Georgia State University
Digital Archive @ GSU

Mathematics Theses | Department of Mathematics and Statistics

8-8-2005

Analyzing the Combination of Polymorphisms Associating with Antidepressant Response by Exact Conditional Test

Baofu Ma
bma1@student.gsu.edu

Follow this and additional works at: http://digitalarchive.gsu.edu/math_theses

Recommended Citation
Ma, Baofu, "Analyzing the Combination of Polymorphisms Associating with Antidepressant Response by Exact Conditional Test" (2005). *Mathematics Theses*. Paper 2.

This Thesis is brought to you for free and open access by the Department of Mathematics and Statistics at Digital Archive @ GSU. It has been accepted for inclusion in Mathematics Theses by an authorized administrator of Digital Archive @ GSU. For more information, please contact digitalarchive@gsu.edu.

Analyzing The Combination of Polymorphisms Associating With Antidepressant Response By Exact Conditional Test

by

Baofu Ma

Under the Direction of Susmita Datta

Abstract

Genetic factors have been shown to be involved in etiology of a poor response to the antidepressant treatment with sufficient dosage and duration. Our goal was to identify the role of polymorphisms in the poor response to the treatment. To this end, 5 functional polymorphisms in 109 patients diagnosed with unipolar, major depressive disorder are analyzed. Due to the small sample size, exact conditional tests are utilized to analyze the contingency table. The data analysis involves: (1) Exact test for conditional independence in a high dimensional contingency table; (2) Marginal independence test; (3) Exact test for three-way interactions. The efficiency of program always limits the application of exact test. The appropriate methods for enumerating exact tables are the key to improve the efficiency of programs. The algorithm of enumerating the exact tables is also introduced.

INDEX WORDS: Antidepressant, Exact test, Contingency, Conditional independence, Marginal independence, Enumerating, High dimensional table

Analyzing The Combination of Polymorphisms Associating With Antidepressant Response By Exact Conditional Test

By

Baofu Ma

A thesis Submitted in Partial Fulfillment of the Requirements for the Degree of

Master of Science

Georgia State University

2005

Analyzing The Combination of Polymorphisms Associating With Antidepressant Response By Exact Conditional Test

By

Baofu Ma

Major Professor: Susmita Datta
Committee: Gengsheng Qin
Yichuan Zhao

Electronic Version Approved:

Office of Graduate Studies
College of Arts and Sciences
Georgia State University
August 2005

ACKNOWLEDGMENTS

First of all, I would like to thank my advisor and committee chair, Dr. Susmita Datta, for guiding and supervising me throughout of this thesis. Thank her for her valuable suggestions whenever I meet the problems. I would like to thank Dr. Gengsheng Qin and Dr. Yichuan Zhao for serving on my thesis committee and helpful suggestion. I must extend my thanks to Dr. Grant for providing the data and Dr. Yang for the great help on programming.

Finally, I would like to thank my wife who always provides encouragement and love to me especially when I meet difficulties. I also express my thanks to my parents for their understanding and encouragement.

TABLE OF CONTENTS

ACKNOWLEDGMENTS iv

LIST OF TABLES vi

CHAPTER I Introduction 1

CHAPTER II Summary of The Result 4

CHAPTER III Exact Test For Conditional Independence In High Dimensional Contingency Tables 7

CHAPTER IV Marginal Independence Test 12

CHAPTER V Three-Way Interaction Test 13

CHAPTER VI The Algorithm In Enumerating The Tables 16

REFERENCES 19

Appendix I: Common Subroutines 21

Appendix II: S-Plus Code for Exact Conditional Independent Test Subroutine 30

Appendix III: S-Plus Code for Exact Conditional Test Subroutine 33

Appendix IV: S-Plus Code for Conditional Independent Test 35

Appendix V: S-Plus Code for Fisher's Exact Test and Three-way Interaction Test 38

Appendix VI: Antidepressant Treatment Data 40

LIST OF TABLES

Table 1. The percentage of poor response for the different combination of l, s, A, and G **6**

Table 2. Independence test between polymorphisms and response conditioned on all other polymorphisms **10**

Table 3. Independence test between polymorphisms and response conditioned on one polymorphism **11**

Table 4. Marginal independence test between response to treatment and polymorphisms **12**

Table 5. The distribution of w for test three-way interaction MAO-B × 5-HTTLPR × Response **15**

Table 6. The distribution of w for test three-way interaction MAO-B × MAO-A × Response **15**

CHAPTER I

Introduction

A large proportion of depressed patients fail to respond to antidepressant treatments with sufficient dosage and duration, although antidepressants are effective in treating major depression. Previous research showed that the situation of depression patients would become worse if the poor response to the treatment occurred. For instance, Schatzberg (1983) showed that the depressed patients who had failed to respond to the treatment attempted suicide more frequently.

Previous researches have shown that genetic factors contribute to the poor response to the treatment. References include Franchini (1998) and O'Reilly (1994). Our goal was to identify the role of polymorphisms in the poor response. To this end, the following functional polymorphisms are selected as candidates: (1) a 17 bpVNTR within intron 2 of the 5-HTT gene; (2) a 44 bp insertion/deletion polymorphism in the 5-HTT promoter region (5-HTTLPR); (3) an A to G SNP in intron 13 of MAO-B; (4) a 30 bp VNTR within the MAO-A promoter; (5) a C to G SNP in the serotonin receptor 5-HT_{1A} promoter region. Each polymorphism has two type alleles denoted by 5-HTTLPR l/s, VNTR 10/12, MAO-B A/G, 5-HT_{1A} C/G, and MAO-A 3/4.

109 patients with unipolar, major depressive disorder were recruited from Lookout Mountain Community Services Psychiatric Outpatient Clinics Georgia, US. All patients received separate trials of one or more antidepressants with adequate dosages and duration. The severity of depression and acute treatment response were prospectively assessed based on extensive psychiatric interviews and retrospectively assessed based on

an extensive review of all existing medical records. The patients were classified as non-responders if they failed to respond to all antidepressants treatment, as responders if they experienced a full remission under all antidepressants, or as partial responders if they experienced a partial remission under one or more of the antidepressants or experienced a partial or full remission under some of the antidepressants, but fail to others. If the patients initially responded but did not experience a full remission at the end of the study, they were classified as either responder or partial responders based on the degree of their initial response.

The usual hypothesis test in multidimensional contingency tables is based on asymptotic methods. However, the asymptotic methods may be misleading in analyzing contingency tables with the small sample size or a large number of cells with small frequencies. The alternative methods are exact tests that are conditional on a subset of marginal totals. There are many papers referring exact test. Agresti (1992) made a great survey in the exact methods for contingency table. Kreiner (1992) did the further work on exact inference in multidimensional tables. Morgan and Blumenstein (1992) provide a more general method in exact tests for multidimensional contingency tables based on the marginal sum of the reduced model and full model.

In our problem, the $2^5 \times 3$ contingency table with sample size 109 has a large number of cells with zero frequencies or frequencies which are less than 5. The exact conditional test is preferable to analyze this table. However, the running time of programs always limits the applications of the exact tests. The appropriate algorithms in enumerating exact tables can dramatically improve the efficiency of programs. Many

great algorithms have been introduced, such as Cox and Plackett (1980), Morgan and Blumenstein (1991), Vollset et al. (1991), and Agresti (1992). The algorithms will be introduced in the C hapter VI.

CHAPTER II

Summary of The Result

Among 109 patients, we categorized 49 as responder, 46 as partial responder, and 14 as non-responders based on the criteria above. The data set is summarized in appendix table A. As shown in Table A, we categorize 5-HTTLPR l as containing one or more copies of l alleles, so it contains genotypes l/l and l/s. Similarly, frequency of MAO-B A allele contain genotypes A/A and A/G, frequency of MAO-A 4 allele contain genotypes 4/4 and 3/4, frequency of VNTR 10 allele contain genotypes 10/10 and 12/10, and frequency of 5-HT_{1A} C allele contain genotypes C/C and G/C.

First, we are interested in whether one particular polymorphism is independent of the response of antidepressant treatment conditioned on other polymorphisms. Kreiner (1992) referred that the conditional independence tests in multidimensional tables could be converted to the tests of three-way contingency tables. By this idea, the $2^5 \times 3$ contingency table is changed to a $2 \times 3 \times 16$ table. For example, the conditional independence between 5-HTTLPR and Response can be tested by the three-way table LPR*Response*Y where Y is the combination of other four polymorphisms. The test results show that all the polymorphisms are conditional independent of the response to the treatment (all P-values are greater than 0.1). However, some information may be lost due to the large number of zero cells in this $2^5 \times 3$ table. We go to test the independence between polymorphisms and the response conditioned on just one polymorphism. We found that the response to treatment is dependent on the polymorphism 5-HTTLPR conditioned on MAO-B (P-value=0.015) and dependent on the polymorphism MAO-B

conditioned on 5-HTTLPR (P-value=0.005). The P-value (=0.056) for testing the independence between MAO-B and response conditioned on MAO-A is also not too large. This result indicates that the response to treatment is correlated to MAO-B and 5-HTTLPR, but it is not clear whether the interaction between the polymorphisms will affect the response to treatment.

Further study on detecting the relationship among the response to the treatment, MAO-B, 5-HTTLPR, and MAO-A is made. The three-way interaction among the response to the treatment, MAO-B, and 5-HTTLPR is found to be significant with p-value = 0.0255 indicating that the response to treatment associates with the interaction between MAO-B and 5-HTTLPR. Then, all possible combinations of 5-HTTLPR l, 5-HTTLPR s, MAO-B A, and MAO-B G are ranked. Table 1 shows that the combination of l, and A has the highest percentage of poor response. It confirms the previous research of Dr. Datta et al. that the simultaneous presence of 5-HTTLPR l and MAO-B A alleles affords a high level of poor response to antidepressant treatments. If we use Poisson regression to test the three-way interaction MAO-B*5-HTTLPR*Response, it is hard to say whether the three-way interaction is significant or not with P-value=0.053. In testing the three-way interaction of MAO-A, MAO-B, and response to the treatment, the three-way interaction is not statistically significant (P-value=0.279) indicating the effects of MAO-B on the treatment are the same given MAO-A 3 allele presenting or MAO-A 4 allele presenting.

Next, we go to test the effect of individual polymorphism on the response in terms of the marginal independence test between polymorphism and the response to the

treatment. The response is found to be dependent of the MAO-B A/G (P-value=0.0478), and the p-value in testing 5-HTTLPR l/s is 0.0515. Both P-values are around borderline p=0.05.

In summary, the data shows that the poor response to the treatment is associated with the polymorphisms MAO-B and 5-HTTLPR. The individual tests for the two polymorphisms show a borderline value p=0.05. However, when alleles MAO-B A and 5-HTTLPR l simultaneously present in the depressed patients, the patients show a high percentage of poor response. The P-value 0.025 for this interaction effect is quite small.

Table1. The percentage of poor response for the different combination of l, s, A, and G

Allele combination	Frequency			Percentage	
	R	P	N	R	P+N
l, A	20	32	13	30.8%	69.2%
s, G	3	4	0	42.9%	57.1%
s, A	11	5	0	68.8%	31.2%
l, G	15	5	1	71.4%	28.6%

R: response P: partial response N: non-response

CHAPTER III

Exact Test for Conditional Independence In High Dimensional Contingency Tables

At the first glance, the exact test for $2^5 \times 3$ is almost impossible. However, Kreiner (1992) shows that a conditional independence test in a multidimensional table can be converted to a conditional independence test in a three-way contingency table. In our problem, all the five polymorphisms, denoted by A, B, C, D, and E, have 2 levels. The response, denoted by R, has 3 levels. The hypothesis test of A $\perp$ R | BCDE is equivalent to the test of the loglinear model (ABCDE, RBCDE) against the saturated model (ABCDER). Defined Y=BCDE, our problem goes to test the loglinear model (AY, RY) against (ARY) where Y has 16 levels.

Agresti (1992) shows that an efficient score statistic for the test of conditional independence is Pearson statistic, $X^2 = \sum_k \chi_k^2$. χ_k^2 is the Pearson statistic which test the independence between A and R in the kth partial table given the row total and column total are fixed . χ_k^2 turns out to be

$$\chi_k^2 = \sum_i \sum_j (n_{ijk} - \hat{m}_{ijk})^2 / \hat{m}_{ijk}$$

where $\hat{m}_{ijk} = n_{i+k} n_{+jk} / n_{++k}$. In exact test, one can systematically generate all the tables that satisfy the fixed row totals and column totals in each partial table. For each pseudo table, we calculate a Pearson statistic X^2. The exact distribution of X^2 can be got based on the set of the pseudo tables. The P-value is the probability that the X^2 value of pseudo

tables is greater than or equal to the X^2 value of the observed table from the exact distribution of X^2.

Because there is no the interaction term AR in the reduced model (AY, RY), we can treat the k partial tables as independent of each other under null hypothesis. The probability of a pseudo table is $P = \prod_k P_k$, where P_k is the probability of the kth partial table. Agresti (1992) showed that in a $I \times J$ table, the distribution of $\{n_{ij}\}$ turned out to be multiple hypergeometric given the row total and column total. Hence, P_k can be calculated by the following equation:

$$P_k = \frac{(\prod_i n_{i+k}!)(\prod_j n_{+jk}!)}{n_{++k}!\prod_i\prod_j n_{ijk}!}$$

The P-value for conditional independence test is the summation of probabilities for the tables whose Pearson statistics are greater than or equal to the observed table denoted by:

$$P - value = \sum_{X^2 \geq X^2_{obs}} P ,$$

where X^2_{obs} is the observed Pearson statistic and $P = \prod_k P_k$ denotes the probability of the three-way pseudo tables. Because there are too many exact tables of $\{n_{ijk}\}$, the probability of each table is very small. To ensure the accuracy in calculating the probability and avoid the computer overflow, P_k is replaced by P_k^*, where $P_k^* = 10P_k$. The P-value is calculated by $(\sum_{X^2 \geq X^2_{obs}} P^*)/10^k$, where $P^* = \prod_k P_k^*$.

For large sample size, each partial table follows asymptotic Chi-square distribution with (I-1)×(J-1) degree of freedom. Considering the independence of the K partial tables, the Pearson statistic $X^2 = \sum_k \chi_k^2$ follows asymptotic Chi-square distribution with (I-1)×(J-1) ×K degree of freedom. Base on the asymptotic Chi-square distribution, the P-value for testing conditional independent is Pr($X^2 \geq X_{obs}^2$).

The test result is shown in Table 2. The comparison between the exact test and asymptotic test is also shown. The asymptotic test shows that each of the polymorphisms are independent of the response to the treatment conditioned on the other polymorphisms. The exact test also fails to find out any dependence between polymorphisms and response (all P-values are greater than 0.1). However, some information may be lost due to the large number of zero cells in this $2^5 \times 3$ table. We can also see that the P-values are quite different between the exact test and asymptotic test. It convinces that the asymptotic test is not appropriate in this data set.

Since some dependence relationships may fail to be detected due to the extremely small frequencies in the 2 × 3 × 16 table, we go to test the independence between polymorphisms and response conditioned on just one polymorphism. The result is shown in Table 3. Table 3 shows that the response to treatment is dependent of the polymorphism 5-HTTLPR conditioned on MAO-B (P-value=0.015) and dependent of the polymorphism MAO-B conditioned on 5-HTTLPR (P-value=0.005). The P-value (=0.056) for testing the independence between response and MAO-B conditioned on MAO-A is close to the borderline 0.05. Hence, the further study is necessary on the

relationship among the polymorphisms MAO-B, 5-HTTLPR, and the response to treatment and the relationship among the polymorphisms MAO-B, MAO-A, and the response to treatment. Compare the P-values of exact test to asymptotic test in table 3, we can see that the exact distribution of Pearson statistic is close to Chi-square distribution with 4 degree of freedom.

This result indicates that the response to treatment is correlated to MAO-B and 5-HTTLPR, but it is not clear whether the interaction between the polymorphisms will affect the response to treatment. A test of three-way interaction among response, MAO-B, and 5-HTTLPR will make more sense.

Table 2. Independence test between polymorphisms and response conditioned on all other polymorphisms

Polymorphism	X^2_{obs}	p-value	
		Exact	Asymptotic
VNTR	14.613	0.744	0.996
5-HTTLPR	18.1412	0.171	0.977
MAO-B	20.6343	0.202	0.939
MAO-A	9.339	0.957	0.99997
5-HT1A	11.571	0.697	0.9997

VNTR: a 17 bpVNTR within intron 2 of the 5-HTT gene;

5-HTTLPR: a 44 bp insertion/deletion polymorphism in the 5-HTT promoter region;

MAO-B: an A to G SNP in intron 13 of MAO-B;

MAO-A: a 30 bp VNTR within the MAO-A promoter;

5-HT1A: a C to G SNP in the serotonin receptor 5-HT_{1A} promoter region

Table 3. Independence test between polymorphisms and response conditioned on one polymorphism

Condition on / Polymorphism		VNTR	5-HTTLPR	MAO-B	MAO-A	5-HT1A
VNTR	X^2_{obs}	NA	0.500	2.736	0.900	1.601
	P-Exact	NA	0.967	0.662	0.923	0.835
	P-Asy	NA	0.974	0.603	0.925	0.809
5-HTTLPR	X^2_{obs}	5.536	NA	11.763	7.060	4.714
	P-Exact	0.255	NA	0.015	0.147	0.332
	P-Asy	0.237	NA	0.019	0.133	0.318
MAO-B	X^2_{obs}	8.250	12.466	NA	9.669	8.019
	P-Exact	0.083	0.005	NA	0.056	0.091
	P-Asy	0.083	0.014	NA	0.046	0.091
MAO-A	X^2_{obs}	0.431	1.770	3.494	NA	2.791
	P-Exact	0.988	0.634	0.480	NA	0.647
	P-Asy	0.980	0.778	0.479	NA	0.593
5-HT1A	X^2_{obs}	3.166	1.919	3.613	4.561	NA
	P-Exact	0.548	0.605	0.482	0.371	NA
	P-Asy	0.530	0.751	0.461	0.335	NA

VNTR: a 17 bpVNTR within intron 2 of the 5-HTT gene;

5-HTTLPR: a 44 bp insertion/deletion polymorphism in the 5-HTT promoter region;

MAO-B: an A to G SNP in intron 13 of MAO-B;

MAO-A: a 30 bp VNTR within the MAO-A promoter;

5-HT1A: a C to G SNP in the serotonin receptor 5-HT_{1A} promoter region

P-Exact: the P-value by exact test P-Asy: the P-value by asymptotic Chi-square test

CHAPTER IV

Marginal Independence Test

In many cases, the conditional independence does not indicate the marginal independence. To avoid losing some dependence information between response and polymorphisms, it is necessary to test the marginal independence. Since our marginal table still contains some small frequency cells, Fisher's exact test is utilized in this problem. The test result shown in Table 4 indicates that the response to the treatment is dependent of the MAO-B A/G (P-value=0.0478), and the p-value in testing 5-HTTLPR l/s is also not too large (p-value=0.0515). The more general exact test method introduced in next chapter also gets the same result to Fisher's exact test. Comparing the test result between Fisher's test and asymptotic Chi-square test, we notice that the P-value of Fisher's test in testing 5-HTTLPR does not agree with the asymptotic test.

Table 4. Marginal independence test between response to treatment and polymorphisms

Polymorphism	p-value	
	Fisher	Asymptotic
VNTR	0.7731	0.787
5-HTTLPR	0.0515	0.085
MAO-B	0.0478	0.032
MAO-A	1.00	0.904
5-HT1A	0.3228	0.342

VNTR: a 17 bpVNTR within intron 2 of the 5-HTT gene;

5-HTTLPR: a 44 bp insertion/deletion polymorphism in the 5-HTT promoter region;

MAO-B: an A to G SNP in intron 13 of MAO-B;

MAO-A: a 30 bp VNTR within the MAO-A promoter;

5-HT1A: a C to G SNP in the serotonin receptor 5-HT_{1A} promoter region

CHAPTER V

Three-Way Interaction Test

Morgan and Blumenstein (1992) provide a more general method in exact tests for multidimensional contingency tables. This method is conditioned on the marginal sum of the reduced model. Given contingency table x, denote *M(x, F)* as the set of marginal sums of full model, *M(x, R)* as the set of marginal sums of reduced model, and $w = M(x, H)$ as the set of marginal sums that are in the full model but not in the reduced model. The set of marginal sums based on observed table, denoted as $w^* = M(x^*, H)$, can be used as the test statistic of the reduced model against the Full model. The distribution of w can be found by enumerating the possible tables conditioned on the observed set of marginal sums of reduced model. The probability of w is:

$$\mathrm{P}(W = w;\ F \mid x \in T) = \sum_{U(w)} P(x;F) / \sum_{T} P(x;F)$$

where T is the set of all possible tables conditioned on the observed marginal sums of the reduced model, *U(w)* is the set of the tables that *M(x, H)* equal to w in T. *P(x; F)* is the probability of table x in the multinomial sampling.

$$P(X = x;F) = n!c(x)\prod_i P_i^{x_i}$$

where n is the marginal total, P_i is the probability of the *ith* cell, x_i is the count of the *ith* cell, and $c(x) = \prod_i 1/x_i!$. Under null hypothesis, the probability of w can be reduced to:

$$P(W = w;\ F|x \in T) = \sum_{U(w)} c(x) / \sum_{T} c(x)$$

The P-value in testing the reduced model against the full model is the probability of w satisfied $P(w) \leq P(w^*)$.

From previous chapter, we notice that the response to the treatment is dependent of MAO-B A/G conditioned on 5-HTTLPR l/s or MAO-A 3/4 and dependent of 5-HTTLPR l/s conditioned on MAO-B. We make further study on these three polymorphisms based on the test statistic above.

First, the three-way interaction is tested based on the data from 5-HTTLPR l/s, response to antidepressant treatment, and MAO-B A/G (2 × 3 × 2 table). The reduced model is [MAO-B * 5-HTTLPR, MAO-B * Response, 5-HTTLPR * Response] against the full model [MAO-B * 5-HTTLPR * Response]. The test result is shown in Table 5. The three-way interaction is found to be significant with p-value = 0.0255. It confirms the previous research of Dr. Datta et al. that the simultaneous presence of 5-HTTLPR l and MAO-B A alleles affords a high level of poor response to antidepressant treatments. We also analyze this data set by Poisson regression model. The model without three-way interaction term yields a AIC value 25.89. Next, a three-way interaction term is added into the model, the AIC value is lowered to 24.00. The model is improved, but it is hard to say whether the improvement is significant or not with P-value = 0.053.

Next, the reduce model [MAO-B * MAO-A, MAO-B *Response, MAO-A* Response] against the full model [MAO-B*MAO-A*Response] is tested. The test result

is shown in Table 6. The three-way interaction is found to be insignificant with p-value = 0.279.

Table 5. The distribution of w for test three-way interaction MAO-B × 5-HTTLPR × Response

$P\text{-}W_{obs}$: 0.02436				P-value: 0.0255		
P-W:	0.41092	0.32662	0.17011	0.06682	0.02436	0.00115
	$1.55530*10^{-5}$	$4.08095*10^{-8}$				

$P\text{-}W_{obs}$: The probability of w in observed table
P-W: the probability that statistic W is equal to w in the table set T.

Table 6. The distribution of w for test three-way interaction MAO-B × MAO-A × Response

$P\text{-}W_{obs}$: 0.13025				P-value: 0.279			
P-W:	0.42040	0.30029	0.13025	0.05584	0.04692	0.02509	0.01766
	0.00132	0.00223					

$P\text{-}W_{obs}$: The probability of w in observed table
P-W: the probability that statistic W is equal to w in the table set T.

CHAPTER VI

The Algorithm In Enumerating The Tables

The running time of programs always limits the applications of the exact tests. The appropriate algorithms in enumerating exact tables can dramatically improve the efficiency of programs. The effective way is to find out the cells that are free to vary and find out the range of counts in each cell.

Morgan and Blumenstein (1992) provide an algorithm in enumerating exact tables. In this algorithm, the list of cells that are free to vary can be found by the set of fixed margins that are in the reduced model. Each of the margins can fix a set of cells in the contingency table. The complete set of fixed cells is obtained by going through the set of fixed margins. The remaining cells would be what are designed to be free to vary. The counts systematically change in the cells that are free to vary. The remaining counts can be calculated based on the fixed marginal totals and the counts in the cells that are designed to be free to vary. For example, the reduce model is [AB, AC, BC] against the full model [ABC] in a $2 \times 2 \times 3$ table. The degree of freedom is $(2\text{-}1) \times (2\text{-}1) \times (3\text{-}1) = 2$. Hence, only two cells are designed to be free to vary. The computations are dramatically reduced.

Cox (1980) provides another algorithm in enumerating exact tables. In his algorithm, a table that satisfies the fixed marginal totals, such as the observed table, is specified. Other tables can be generated by adding this table to the pseudo tables with zero marginal totals.

Setting the appropriate range of counts in each cell can dramatically reduce the calculation in counting the exact tables. Given the set of fixed marginal totals, each sell are subjected to a subset of the margins. The count of the specific cell is less than or equal to the minimum of the subset of the margins. Given the marginal totals and the set of upper limit counts, the lower limit count of the specific cell is generated by subtracting the upper limit counts of the other cells that are subjected to the same margins from the marginal totals. For example, consider a 2×3 table. The row totals and column totals are fixed.

$$\begin{array}{|ccc||c|} 4 & 8 & 9 & 21 \\ 1 & 1 & 2 & 4 \\ \hline 5 & 9 & 11 & \end{array}$$

The upper limit of the count for cell a_{11} is equal to min{21,5}=5. Get the upper limit count of the other cells by the same idea. The upper limit count matrix is:

$$\begin{vmatrix} 5 & 9 & 11 \\ 4 & 4 & 4 \end{vmatrix}$$

To calculate the lower limit of the count for cell a_{12}, we subtract the upper limit counts of a_{11} and a_{13} from the row total (21-5-11=5) and subtract the upper limit counts of a_{22} from the column total (9-4=5). Then, the lower limit of the count for cell a_{12} is equal to max{21-5-11, 9-4}=5. The lower limit count matrix is:

$$\begin{vmatrix} 1 & 5 & 7 \\ 0 & 0 & 0 \end{vmatrix}$$

In this example, two cells are free to vary. Set a_{11} and a_{12} as the free cells whose ranges are from 1 to 5, and 5 to 9 separately. We only need generate 25 tables. If we

systematically vary the six cells to get the tables that are subjected to the marginal totals, 5^6 tables are generated and most of them should be discarded. The calculation is dramatically reduced.

Two subroutines that use the above algorithm are provided in this paper: ECIT is used to test conditional independent by the exact test method in chapter 3; ECT using the exact test method in Chapter V is more general.

REFERENCES

[1] Agresti, A. (1992) A Survey of Exact Inference for Contingency Tables. *Statistical Science*, **7**, 131-153.

[2] Agresti, A. (1996) *An Introduction to Categorical Data Analysis*. New York: John Wiley & Sons, Inc.

[3] Cox, M. A. A., Plackett, R. L. (1980). Small Samples in Contingency Tables. *Biometrika*, **67**, 1-13.

[4] Franchini, L., Serretti, A., Gasperini, M., Smeraldi, E. (1998) Familial concordance of fluvoxamine response as a tool for differentiating mood disorder pedigrees. *J. Psychiatr. Res.*, **32**, 255-259.

[5] Haberman, S. J. (1988) A warning on the use of chi-squared statistics with frequency tables with small expected cell counts. *J. Am. Statist. Ass.*, **83**, 555-560.

[6] Kreiner, S. (1992). [A Survey of Exact Inference for Contingency Tables]: Comment: Exact Inference in Multidimensional Tables. *Statistical Science*, **7**, 163-165.

[7] Morgan, W. M., Blumenstein, B. A. (1991). Exact Conditional Tests for Hierarchical Models in Multidimensional Contingency Tables. *Applied Statistics*, **40**, 435-442.

[8] O'Reilly, R. L., Bogue, L., Singh, S. M. (1994) Pharmacogenetic response to antidepressants in a multicase family with affective disorder. *Biol Psychiatry*, **36**, 467-471.

[9] Schatzberg, A. F., Cole, J. O., Cohen, B. M., Altemus, R. I., Sniffin, C. M. (eds). (1983) *Survey of Depressed Patients who have Failed to Respond to Treatment*. Washington, D.C.: American Psychiatric Press.

[10] Vollest, S. E., Hirji, K. F. and Elashoff, R. M. (1991). Fast computation of exact confidence limits for the common odds ratio in a series of 2×2 tables, *J. Amer. Statist. Assoc.* **86**, 404-409.

[11] Yang X., Grant R. E., Datta S. Grant K. B. A combination of serotonin transporter and monoamine oxidase B polymorphisms predicts antidepressant response in major depressive disorder.

Appendix I: Common Subroutines

```
##################################################
#decompose formula into elements
#for example: decompose 'ab*bc' into "ab" "bc"
##################################################

decompose.elem<-function(formula)
{
   elem<-character(0)
   while (regexpr('\\*',formula)!=-1)
   {
      pst<-regexpr('\\*',formula)[1]
      elem <-c(elem,substring( formula , first = 1,  last = pst-1 ))
      formula<-substring( formula , first = pst+1)
   }
   elem<-c(elem,formula)
   return(elem)
}

#############################################################
#compare 2 formulas
#return True if 2 formulas are equal
#otherwise reture False
#for example compare.formulas('ac',"ab*ac") will return False
#############################################################

compare.formulas<-function(formula1,formula2)
{
   temp1<-decompose.elem(formula1)
   n1<-length(temp1)
   temp2<-decompose.elem(formula2)
   n2<-length(temp2)
   if (n1!=n2) return(F)
   rst<-F
   for (i in 1:n1)
   {
      for (j in 1:n2)
      {
         if(temp1[i]==temp2[j]) rst<-T
      }
      if (rst==F) return (F)
      else rst<-F
   }
   return(T)
}

#########################################################
# Test whether a formula is contained in a list of formulas
# Return the subscript if A belongs to B
# Otherwise return 0
# For example, AbelongB('3*2',c("1" ,  "1*2", "2",   "2*3" ,"3")) returns 4.
#########################################################
AbelongB<-function(A,B)
{
   if ((length(B)==0) || (A=='')) return(0)
   for (i in 1:length(B))
   {
```

```
        if (compare.formulas(A,B[i])) return(i)
    }
    return(0)
}

##########################################################
#decompose formular into a list of sub-formulars
#For example decompose.sub ('col1*col2') will decompose
# 'col1*col2' into ('col1', 'col2','col1*col2')
##########################################################

decompose.sub<-function(formular)
{
    while (regexpr(' ',formular)!=-1) #eliminate blank from formula
    {
        pst<-regexpr(' ',formular)[1]
        formular<-paste(substring(formular, first = 1,  last = pst-
    1),substring(formular, first = pst+1),sep='')
    }
    elem1<-character(0)
    while (regexpr('\\*',formular)!=-1)
    {
        pst<-regexpr('\\*',formular)[1]
        if (length(elem1)==0) tmpelm<-paste(elem1,substring(formular, first = 1,
    last = pst-1),sep='')
        else
        {
            tmpelm<-paste(elem1,substring(formular, first = 1,  last = pst-
    1),sep='*')
            tmpelm<-c(tmpelm,substring(formular, first = 1,  last = pst-1))
        }
        elem1 <-c(elem1,tmpelm)
        formular<-substring( formular, first = pst+1)
    }
    if (length(elem1)==0) elem1<-c(elem1,paste(elem1,formular,sep=''))
    else
    {
        elem1<-c(elem1,paste(elem1,formular,sep='*'))
        elem1<-c(elem1,formular)
    }
    return(elem1)
}

#######################################################################
#decompose str and eliminate blank
#for example, decompose.substr(' 1*2 | 2 * 3') returns ('1*2', '2 * 3')
#######################################################################

decompose.substr<-function(str)
{
    elem<-character(0)
    while (regexpr('|',str)!=-1)
    {
        pst<-regexpr('|',str)[1]
        elem <-c(elem,substring(str, first = 1,  last = pst-1 ))
        str<-substring( str, first = pst+1)
    }
    elem<-c(elem,str)
    for (i in 1:length(elem))
    {
```

```
        while (regexpr(' ',elem[i])!=-1) #eliminate blank from formula
        {
            pst<-regexpr(' ',elem[i])[1]
            elem[i]<-paste(substring(elem[i], first = 1,  last = pst-
    1),substring(elem[i], first = pst+1),sep='')
        }
    }
    return(elem)
}

#################################################################
# decompose formula into the list of marginal cells
# for example, decompose.marg.list(y~V3* V5|V2*V5) will generate
# the list of margins ("V3","V3*V5","V5","V2","V2*V5") and the
# the list of variables in the model ("V3" "V5" "V2")
#########################################################
decompose.marg.list<-function(formula)
{
    tmpform1<-as.character(formula[3])
    elem<-decompose.substr(tmpform1)
    rst<-character(0)
    varlist<-character(0)
    for (i in 1:length(elem))
    {

        temp<-decompose.sub(elem[i])
        for (j in 1:length(temp))
        {

            if (AbelongB(temp[j],rst)==0)
            {
                rst<-c(rst,temp[j])
                if (regexpr('\\*',temp[j])==-1) varlist<-c(varlist,temp[j])
            }
        }
    }

    return(list(marglist=rst,varlist=varlist))
}

#################################################################
# construct observed marginal table
# arguments:
# marg.list: the set of marginal totals. For example c('V1', 'V1*V2', 'V2')
# varorder: the ordered dimensions. For example, c('V1','V2') indicates
#            V1 is the first dimension and V2 is the second dimension.
# data: multi-way contingency table
# return: the set of marginal totals
#################################################################
marginal.cell<-function(marg.list,varorder,data)
{
    rst<-sum(data)
    for(i in 1:length(marg.list))
    {
        var.1<-pmatch(decompose.elem(marg.list[i]),varorder)
        rst<-c(rst,apply(data,var.1,sum))
    }
    return(rst)
}
```

```
#######################################################
#Compare whether 'a' match the set of marginal total of a talbe
# a: the vector of a set of marginal totals
# marg.list: the set of margins in the reduced model
#varorder: the dimension list of the observed contingency table
#data: the observed contingency table
#return T if a match the set of margins of table 'data';
#otherwise return F
#######################################################################
compar.margcell<-function(a,marg.list,varorder,data)
{
   p<-2
   for(i in 1:length(marg.list))
   {
      var.1<-pmatch(decompose.elem(marg.list[i]),varorder)
      rst<-apply(data,var.1,sum)
      up<-p+length(rst)-1
      if (!compar(rst,a[p:up])) return(F)
      p<-up+1
   }
   return(T)
}

#####################################################
#function for comparing two vector.
#x, y: the vectors compared
#Return true,if they are equal. otherwise return F
#####################################################
compar<-function(x,y)
{
   if (length(x)!=length(y)) return(F)
   z<-ifelse(x!=y,1,0)
   return(ifelse(max(z)==0,T,F))
}

#####################################################################
# Find out the maximum count table that subject to the reduced model
# marg.list: the set of margins in the reduced model
# var: the variable list of the reduced model
# varorder: the variable list in the observed contingency table
# data: the observed contingency table
# return a vector of the maximum count
#######################################################################

max.cell<-function(marg.list,var,varorder,data)
{
   pmatch1<-pmatch(var,varorder)
   data.dim<-dim(data)
   tab.dim<-data.dim[pmatch1]
   ncell.table<-max(cumprod(tab.dim))   #the number of cells
   rst1<-rep(sum(data),ncell.table)
   for(k in 1:length(marg.list))
   {
      t1<-decompose.elem(marg.list[k])
      t3<-rep(0,length(t1))
      pmatch2<-pmatch(t1,varorder)
      app.dim<-data.dim[pmatch2]
      free.pmatch<-pmatch1[-match(pmatch2,pmatch1)]
      free.dim<-data.dim[free.pmatch]
      cumpr<-c(1,cumprod(app.dim))
```

```
        cumpr<-cumpr[-length(cumpr)]
        var.1<-pmatch(t1,varorder)
        t2<-apply(data,var.1,sum)
        for (i in 1:length(t2))
        {
            temp1<-rep(1,length(var))
            ppi<-1
            while(ppi<=length(free.pmatch))
            {
                n<-length(cumpr)
                no.cell<-i
                for (j in 1:n)
                {
                    temp1[pmatch(t1[n-j+1],var)]<-floor(no.cell/cumpr[n-j+1])+1
                    no.cell<-no.cell-cumpr[n-j+1]*(temp1[pmatch(t1[n-j+1],var)]-1)
                    if (no.cell==0)
                    {
                    temp1[pmatch(t1[n-j+1],var)]<-temp1[pmatch(t1[n-j+1],var)]-1
                    no.cell<-cumpr[n-j+1]
                    }
                }
                tmpdim<-(tab.dim)
                cumpr1<-c(1,cumprod(tmpdim))
                cumpr1<-cumpr1[-length(cumpr1)]
                posi<-sum((temp1-1)*cumpr1)+1
                cond<-rst1[posi]
                if (cond>t2[i]) rst1[posi]<-t2[i]

                ppi<-1
                temp1[free.pmatch[ppi]]<-temp1[free.pmatch[ppi]]+1
                while(temp1[free.pmatch[ppi]]>tab.dim[free.pmatch[ppi]])
                {
                    temp1[free.pmatch[ppi]]<-1
                    ppi<-ppi+1
                    if (ppi<=length(free.pmatch)) temp1[free.pmatch[ppi]]<-
    temp1[free.pmatch[ppi]]+1
                    else break
                }
            }
        }
    }
    return(rst1)
}

#######################################################################
# find out whether the simulated statistic W is contained in a list of W
# b: the vector of the statistic W
# rst: the set of the statistic W
# wnumber: the number of the statistic W in 'rst'.
# return the position of 'b' in rst if b is contained in 'rst'.
# otherwise return 0.
#######################################################################

WBelong<-function(b,rst,wnumber)
{
    n<-length(b)
    for (wid in 1:wnumber)
    {
        if (compar(b,rst[((wid-1)*n+1):(wid*n)])) return(wid)
    }
```

```
    return(0)
}

##############################################################
# Find out the cells that are free to vary
# marg.list: the set of margins in the reduced model
# varorder: the variable list of the observed table
# dima: The vector of the dimension of table. for example,
#       dima=[2,3] indicates a 2×3 table.
# return the list of cells that are free to change and the
#        the list of fixed cells.
##############################################################

gettab.ab<-function(marg.list,varorder,dima)
{
    #get the fixed cell number
    fix.n<-1
    for (i2 in 1:length(marg.list))
    {
        elem.d<-decompose.elem(marg.list[i2])
        var.2<-pmatch(elem.d,varorder)
        fix.n<-fix.n+max(cumprod(dima[var.2]-1))
        if (i2==1)
        {
            list.t<-marg.list[i2]
            x.t<-length(elem.d)
        }
        else
        {
            pp<-match(T,x.t<length(elem.d))
            if (is.na(pp))
            {
                list.t<-c(list.t,marg.list[i2])
                x.t<-c(x.t,length(elem.d))
            }
            else if (pp==1)
            {
                list.t<-c(marg.list[i2],list.t)
                x.t<-c(length(elem.d),x.t)
            }
            else
            {
                list.t<-c(list.t[1:(pp-
    1)],marg.list[i2],list.t[pp:length(list.t)])
                x.t<-c(x.t[1:(pp-1)],length(elem.d),x.t[pp:length(x.t)])

            }
        }
    }

    wnumber<-0
    rsta<-rstb<-rst<-numeric(0)
    i1<-1
    while(wnumber<fix.n)
    {
        dima1<-dima
        dima2<-dima+1
        var.1<-pmatch(decompose.elem(list.t[i1]),varorder)
        i1<-i1+1
```

```
        f.dima<-dima[var.1]
        cumpr<-c(1,cumprod(f.dima))
        cumpr<-cumpr[-length(cumpr)]

        for (i in 1:max(cumprod(f.dima)))
        {
               n<-length(f.dima)
               no.cell<-i
               for (j in 1:n)
               {
                  dima2[var.1[n-j+1]]<-dima1[var.1[n-j+1]]<-
    floor(no.cell/cumpr[n-j+1])+1
                  no.cell<-no.cell-cumpr[n-j+1]*(dima1[var.1[n-j+1]]-1)
                  if (no.cell==0)
                  {
                     dima2[var.1[n-j+1]]<-dima1[var.1[n-j+1]]<-dima1[var.1[n-
    j+1]]-1
                     no.cell<-cumpr[n-j+1]
                  }
               }
               if (wnumber==0 || WBelong(dima1,rst,wnumber)==0)
               {
                  rst<-c(rst,dima1)
                  rstb<-c(rstb,dima2)
                  wnumber<-wnumber+1

                  if (wnumber==fix.n) break
               }

        }
    }
    #get cells that free to vary
    dima3<-rep(1,length(dima))
    cumpr<-c(1,cumprod(dima))
    cumpr<-cumpr[-length(cumpr)]
    for (k in 1:max(cumprod(dima)))
    {
        n<-length(dima)
        no.cell<-k
        for (j in 1:n)
        {
           dima3[n-j+1]<-floor(no.cell/cumpr[n-j+1])+1
           no.cell<-no.cell-cumpr[n-j+1]*(dima3[n-j+1]-1)
           if (no.cell==0)
           {
              dima3[n-j+1]<-dima3[n-j+1]-1
              no.cell<-cumpr[n-j+1]
           }
        }
        if (WBelong(dima3,rst,wnumber)==0)
        {
           rsta<-c(rsta,dima3)
        }
    }
    return(list(cell.free=rsta, cell.fix=rstb))
}

#######################################################################
# find out the range of cell counts subjected to the reduced model.
# obs.t: the observed table
```

```
# tab.a: the list of cells that are free to vary.
# tab.b: the list of cells that are fixed.
# maxcount: the vector of the maximum cell counts.
# return the range of counts that are allowed to vary
#######################################################################

max.min<-function(obs.t,tab.a,tab.b,maxcount)
{
   dima<-dim(obs.t)
   max.t<-maxcount-as.vector(obs.t)
   min.t<--as.vector(obs.t)
   a.max<-numeric(0)
   a.min<-numeric(0)
   cumpr1<-c(1,cumprod(dima))
   cumpr1<-cumpr1[-length(cumpr1)]
   var.n<-length(dima)
   for (i1 in 1:(length(tab.a)/var.n))
   {
      x<-tab.a[((i1-1)*var.n+1):(i1*var.n)]
      for (i in 1:(length(tab.b)/var.n))
      {
         temp1<-tab.b[((i-1)*var.n+1):(i*var.n)]
         kx<-(x==temp1)
         kx1<-temp1[F==kx]>dima[F==kx]
         if (is.na(match(F,kx1)))
         {
            kx.1<-dima[!kx]
            t.sum<-0
            cumpr<-c(1,cumprod(kx.1))
            cumpr<-cumpr[-length(cumpr)]
            kx.length<-length(kx.1)
            for (j in 1:max(cumprod(kx.1)))
            {
               no.cell<-j
               kxk<-kx
               for (k in 1:kx.length)
               {
                  p<-max(pmatch(rep(F,length(kxk[kxk==F])),kxk))
                  temp1[p]<-floor(no.cell/cumpr[kx.length-k+1])+1
                  no.cell<-no.cell-cumpr[kx.length-k+1]*(temp1[p]-1)
                  if (no.cell==0)
                  {
                     temp1[p]<-temp1[p]-1
                     no.cell<-cumpr[kx.length-k+1]
                  }
                  kxk[p]<-T
               }
               t.sum<-t.sum+max.t[sum(cumpr1*(temp1-1))+1]
            }
            p1<-sum(cumpr1*(x-1))+1
            cond<--(t.sum-max.t[p1])
            if (cond>min.t[p1]) min.t[p1]<-cond
         }
      }
      a.max<-c(a.max,max.t[sum(cumpr1*(x-1))+1])
      a.min<-c(a.min,min.t[sum(cumpr1*(x-1))+1])
   }
   return(list(a.max=a.max,a.min=a.min))
}
```

```
#########################################################################
# generate an exact table that are subjected to the fixed marginal totals
#        given the counts in cells that are free to vary.
# dima: The vector of the dimension of table. for example,
#       dima=[2,3] indicates a 2×3 table.
# tab.a: the list of cells that are free to vary.
# tab.b: the list of cells that are fixed.
# a.count: the counts in cells that are free to vary.
# return the simulated table.
#########################################################################

gettable<-function(dima,tab.a,a.count,tab.b)
{
   var.n<-length(dima)
   while (length(tab.b)>0)
   {
      n<-length(a.count)
      tab.b.t<-tab.b[1:(var.n)]
      t1<-tab.b.t>dima
      t2<-(!t1)*tab.b.t
      needed.n<-max(cumprod(tab.b.t[t1]-1))-1
      t.count<-0
      for (i in 1:n)
      {

         t3<-tab.a[((i-1)*var.n+1):(i*var.n)]*(!t1)
         if (compar(t2,t3))
         {
            if (needed.n==1)
            {
               a.count<-c(a.count,t.count-a.count[i])
               tab.b.t[tab.b.t>dima]<-tab.b.t[(tab.b.t>dima)]-1
               tab.a<-c(tab.a, tab.b.t)
               tab.b<-tab.b[-c(1:var.n)]
               break
            }
            else if (i==n)
            {
               tab.b<-c(tab.b,tab.b[1:var.n])
               tab.b<-tab.b[-c(1:var.n)]
            }
            else
            {
               needed.n<-needed.n-1
               t.count<-t.count-a.count[i]
            }
         }
         else if (i==n)
         {
            tab.b<-c(tab.b,tab.b[1:var.n])
            tab.b<-tab.b[-c(1:var.n)]
         }
      }
   }
   y<-c(1,dima[-(length(dima))])
   indx<-matrix((tab.a-1),ncol=var.n,byrow=T)%*%cumprod(y)+1
   rst<-a.count[order(indx)]
   return(rst)
}
```

Appendix II: S-Plus Code for Exact Conditional Independent Test Subroutine (Used In Chapter III)

```
#################################################################
# Calculate the Pearson's statistic given a I×J table.
# simulat.t: a I×J table.
# return the probability of this table and the Pearson's statistic
#################################################################
pearson.st<-function(simulat.t)
{
   if (sum(simulat.t)==0)
   {
      prob1<-1
      xsq<-0
   }
   else
   {
      io<-apply(simulat.t,1,sum)
      oj<-apply(simulat.t,2,sum)
      t.p<-c(factorial(io),factorial(oj))/factorial(simulat.t[1:5])
      prob1<-cumprod(t.p)[5]/factorial(sum(simulat.t))/factorial(simulat.t[6])
      mij<-outer(io,oj)/sum(simulat.t)
      simulat.t<-simulat.t[mij>0]
      mij<-mij[mij>0]
      xsq<-t(as.vector(simulat.t-mij))%*%(as.vector(simulat.t-mij)/mij)

   }
   return(list(prob=prob1,xsq=xsq))
}


#######################################################################
# The conditional independent test by exact method
# tbl: the tested table. Test the independence between first and second
#      dimension conditioned on the rest dimensions.
# return a list of values:
#      1. Observed Pearson's statistic
#      2. The P-value for independent test
#      3. the number of exact tables subjected to null hypothesis
#######################################################################


ECIT<-function(tbl)
{
   formula.f<-y~x1*x2
   reduce<-'x1*x2'
   dim<-c('x1','x2')
   list1<-decompose.marg.list(formula.f)
   list2<-decompose.substr(reduce)      #reduced part
   rr<-AbelongB(list2,list1$marglist)
   list1$marglist<-list1$marglist[-rr]  #reduced model

   xsq<-numeric(0)
   prb<-numeric(0)
   n16<-numeric(0)
   obs.xsq<-0
```

```
npartial<-cumprod(dim(tbl)[-c(1,2)])
npa<-npartial[length(npartial)]
dpartial<-cumprod(dim(tbl)[c(1,2)])
dpa<-dpartial[2]

for(ip in 1:npa)
{
   obs.t<-as.vector(tbl)[((ip-1)*dpa+1):(ip*dpa)]
   obs.t<-matrix(obs.t,ncol=dim(tbl)[2])
   obs.xsq<-pearson.st(obs.t)$xsq+obs.xsq

   maxcount<-max.cell(list1$marglist,list1$varlist,dim,obs.t)
   dima<-dim(tbl)[c(1:2)]
   tab<-gettab.ab(list1$marglist,dim,dima)
   max.c<-max.min(obs.t,tab$cell.free,tab$cell.fix,maxcount)
   maxrange<-max.c$a.max-max.c$a.min
   n<-length(maxrange)
   t.vector<-rep(0,n)
   t.number<-0
   xsq1<-numeric(0)
   prb1<-numeric(0)
   while (t.vector[n]<=maxrange[n])
   {
      tt<-t.vector[1]<-t.vector[1]+1
      i<-1
      while (tt==(maxrange[i]+1) && i<n)
      {
         t.vector[i]<-0
         tt<-t.vector[i+1]<-t.vector[i+1]+1
         i<-i+1
      }
      #do sth
      if (t.vector[n]>maxrange[n]) rst1<-rep(0,n)+max.c$a.min
      else rst1<-t.vector+max.c$a.min
      rst2<-gettable(dima,tab$cell.free,rst1,tab$cell.fix)+as.vector(obs.t)
      if (is.na(match(T,rst2<0)))
      {
         rst2<-matrix(rst2,ncol=3)
         statis<-pearson.st(rst2)
         t.number<-t.number+1
         xsq1<-c(xsq1,statis$xsq)
         prb1<-c(prb1,statis$prob)
      }
   }

   z<-data.frame(xsq1,prb1)
   z1<-sort.col(z,c('xsq1','prb1'),'xsq1',F)
   if (ip==1)
   {
      xsq<-c(xsq,z1$xsq1)
      prb<-c(prb,z1$prb1)
      n16<-c(n16,t.number)
   }
   else
   {
      pp<-match(T,t.number>n16)
      if (is.na(pp))
      {
         n16<-c(n16,t.number)
```

```
            xsq<-c(xsq,z1$xsq1)
            prb<-c(prb,z1$prb1)
        }
        else if (pp==1)
        {
            n16<-c(t.number,n16)
            xsq<-c(z1$xsq1,xsq)
            prb<-c(z1$prb1,prb)
        }
        else
        {
            xsq<-c(xsq[1:sum(n16[1:(pp-1)])],z1$xsq1,xsq[(sum(n16[1:(pp-
1)])+1):length(xsq)])
            prb<-c(prb[1:sum(n16[1:(pp-1)])],z1$prb1,prb[(sum(n16[1:(pp-
1)])+1):length(prb)])
            n16<-c(n16[1:(pp-1)],t.number,n16[pp:length(n16)])
        }
    }
}


n.lth<-length(n16[n16>1])
tem<-rep(1,n.lth)
tem[1]<-0
flex<-c(0,cumsum(n16[n16>1])[-n.lth])
pvalue<-0
while (pvalue<1*10^n.lth)
{
    tt<-tem[1]<-tem[1]+1
    i<-1
    while (tt==(n16[i]+1) && i<n.lth)
    {
        tem[i]<-1
        tt<-tem[i+1]<-tem[i+1]+1
        i<-i+1
    }
    if ((tem[n.lth]>n16[n.lth])) break
    if(sum(xsq[tem+flex])>=obs.xsq)
    {
        pvalue<-pvalue + cumprod(prb[(tem+flex)]*10)[n.lth]
    }
    else
    {
        temi<-1
        while (tem[temi]==1) {temi<-temi+1}
        tem[1:temi]<-n16[1:temi]
    }
}
return(list(obs.xsq=obs.xsq,pvalue=(pvalue/(10^n.lth)),n.partial=n16))
}
```

Appendix III: S-Plus Code for Exact Conditional Test Subroutine (Used In Chapter V)

```
#################################################################
# perform the exact conditional test
# tbl: the observed table
# dim: the list of variables ordered by the dimension NO.
# formula.f: the full model
# reduce: the reduced term for example, to test A*B+A*C+B*C against
#         the saturated model, the reduced part is A*B*C
# return a list of values, including:
#        1. The observed W statistic
#        2. The distribution of the W statistic
#        3. The number of exact tables that satisfy the fix marginal total.
#        4. The P-value
#################################################################

ect<-function(tbl,dim,formula.f,reduce)
{
   list1<-decompose.marg.list(formula.f)
   list2<-decompose.substr(reduce)      #reduced part
   rr<-AbelongB(list2,list1$marglist)
   list1$marglist<-list1$marglist[-rr]  #reduced model
   maxcount<-max.cell(list1$marglist,list1$varlist,dim,tbl)
   obs.t<-apply(tbl,pmatch(list1$varlist,dim),sum)

   dima<-dim(obs.t)
   tab<-gettab.ab(list1$marglist,dim,dima)
   max.c<-max.min(obs.t,tab$cell.free,tab$cell.fix,maxcount)
   maxrange<-max.c$a.max-max.c$a.min
   n<-length(maxrange)
   t.vector<-rep(0,n)
   rst<-numeric(0)
   wnumber<-0
   while (t.vector[n]<=maxrange[n])
   {
      tt<-t.vector[1]<-t.vector[1]+1
      i<-1
      while (tt==(maxrange[i]+1) && i<n)
      {
         t.vector[i]<-0
         tt<-t.vector[i+1]<-t.vector[i+1]+1
         i<-i+1
      }
      #do sth
      if (t.vector[n]>maxrange[n]) rst1<-rep(0,n)+max.c$a.min
      else rst1<-t.vector+max.c$a.min
      rst2<-gettable(dima,tab$cell.free,rst1,tab$cell.fix)+as.vector(obs.t)
      ####
      if (is.na(match(T,rst2<0)))
      {
            cstar.1<-
   cumprod(factorial(as.vector(obs.t))/factorial(rst2))[length(obs.t)]
            dim(rst2)<-dima
            b<-marginal.cell(list2,dim,rst2)
            if (wnumber==0)
            {
```

```
                cstar.w<-cstar.1
                cstar.t<-cstar.1
                t.number<-1
                rst<-b
                wnumber<-wnumber+1
            }
            else
            {
                cstar.t<-cstar.t+cstar.1
                t.number<-t.number+1
                Wposi<-WBelong(b,rst,wnumber)
                if (Wposi==0)
                {
                       rst<-c(rst,b)
                       cstar.w<-c(cstar.w,cstar.1)
                       wnumber<-wnumber+1

                }
                else
                {
                   cstar.w[Wposi]<-cstar.w[Wposi]+cstar.1
                }
            }
        }
    }
    distr.w<-cstar.w/cstar.t
    c<-marginal.cell(list2,dim,tbl)
    obs.w<-distr.w[WBelong(c,rst,wnumber)]
    return(list(obs.w=obs.w,distr.w=distr.w,t.number=t.number,pvalue=sum(distr.w
    [distr.w<=obs.w])))
}
```

Appendix IV: S-Plus Code for Conditional Independent Test (Exact Test and Asymptotic Test)

```
dat<-read.table("c:\\matemp\\project\\data.txt",header=F)
#MAO-B conditioned on all other polymorphisms
tbl<-table(dat$V5,dat$V2,dat$V3,dat$V4,dat$V6,dat$V7)
tbl<-tbl[,-c(1,2),,,-1,-1]
ECI(tbl)

#LPR conditioned on all other polymorphisms
tbl<-table(dat$V3,dat$V2,dat$V5,dat$V4,dat$V6,dat$V7)
tbl<-tbl[,-c(1,2),,,-1,-1]
ECI(tbl)

#VNTR conditioned on all other polymorphisms
tbl<-table(dat$V4,dat$V2,dat$V5,dat$V3,dat$V6,dat$V7)
tbl<-tbl[,-c(1,2),,,-1,-1]
ECI(tbl)

#HT1A conditioned on all other polymorphisms
tbl<-table(dat$V6,dat$V2,dat$V5,dat$V4,dat$V3,dat$V7)
tbl<-tbl[-1,-c(1,2),,,,-1]
ECI(tbl)

#MAO-A conditioned on all other polymorphisms
tbl<-table(dat$V7,dat$V2,dat$V5,dat$V4,dat$V3,dat$V7)
tbl<-tbl[-1,-c(1,2),,,,-1]
ECI(tbl)

#VNTR conditioned on all other polymorphisms
1-pchisq(14.613,32)
#LPR conditioned on all other polymorphisms
1-pchisq(18.1412,32)
#MAO-B conditioned on all other polymorphisms
1-pchisq(20.6343,32)
#HT1A conditioned on all other polymorphisms
1-pchisq(11.571,32)
#MAO-A conditioned on all other polymorphisms
1-pchisq(9.339,32)

#mao-b on VNTR
tbl<-table(dat$V5,dat$V2,dat$V4)
tbl<-tbl[,-c(1,2),]
ECIT(tbl)

#mao-b on LPR
tbl<-table(dat$V5,dat$V2,dat$V3)
tbl<-tbl[,-c(1,2),]
ECIT(tbl)

#mao-b on HT1A
tbl<-table(dat$V5,dat$V2,dat$V6)
tbl<-tbl[,-c(1,2),-1]
ECIT(tbl)

#mao-b on MAO-A
```

```
tbl<-table(dat$V5,dat$V2,dat$V7)
tbl<-tbl[,-c(1,2),-1]
ECIT(tbl)

#LPR on VNTR
tbl<-table(dat$V3,dat$V2,dat$V4)
tbl<-tbl[,-c(1,2),]
ECIT(tbl)

#LPR on MAO-B
tbl<-table(dat$V3,dat$V2,dat$V5)
tbl<-tbl[,-c(1,2),]
ECIT(tbl)

#LPR on HT1A
tbl<-table(dat$V3,dat$V2,dat$V6)
tbl<-tbl[,-c(1,2),-1]
ECIT(tbl)

#LPR on MAO-A
tbl<-table(dat$V3,dat$V2,dat$V7)
tbl<-tbl[,-c(1,2),-1]
ECIT(tbl)

#VNTR on MAO-B
tbl<-table(dat$V4,dat$V2,dat$V5)
tbl<-tbl[,-c(1,2),]
ECIT(tbl)

#VNTR on LPR
tbl<-table(dat$V4,dat$V2,dat$V3)
tbl<-tbl[,-c(1,2),]
ECIT(tbl)

#VNTR on HT1A
tbl<-table(dat$V4,dat$V2,dat$V6)
tbl<-tbl[,-c(1,2),-1]
ECIT(tbl)

#VNTR on MAO-A
tbl<-table(dat$V4,dat$V2,dat$V7)
tbl<-tbl[,-c(1,2),-1]
ECIT(tbl)

#HT1A on MAO-B
tbl<-table(dat$V6,dat$V2,dat$V5)
tbl<-tbl[-1,-c(1,2),]
ECIT(tbl)

#HT1A on LPR
tbl<-table(dat$V6,dat$V2,dat$V3)
tbl<-tbl[-1,-c(1,2),]
ECIT(tbl)

#HT1A on VNTR
tbl<-table(dat$V6,dat$V2,dat$V4)
tbl<-tbl[-1,-c(1,2),]
ECIT(tbl)

#HT1A on MAO-A
```

```
tbl<-table(dat$V6,dat$V2,dat$V7)
tbl<-tbl[-1,-c(1,2),-1]
ECIT(tbl)

#MAO-A on MAO-B
tbl<-table(dat$V7,dat$V2,dat$V5)
tbl<-tbl[-1,-c(1,2),]
ECIT(tbl)

#MAO-A on LPR
tbl<-table(dat$V7,dat$V2,dat$V3)
tbl<-tbl[-1,-c(1,2),]
ECIT(tbl)

#MAO-A on VNTR
tbl<-table(dat$V7,dat$V2,dat$V4)
tbl<-tbl[-1,-c(1,2),]
ECIT(tbl)

#MAO-A on HT1A
tbl<-table(dat$V7,dat$V2,dat$V6)
tbl<-tbl[-1,-c(1,2),-1]
ECIT(tbl)

1-pchisq(0.5,4) #VNTR on LPR
1-pchisq(2.736,4) #VNTR on MAO-B
1-pchisq(0.9,4)  #VNTR on MAO-A
1-pchisq(1.601,4)  #VNTR on HT1A
1-pchisq(5.536,4)  #LPR on VNTR
1-pchisq(11.763,4)  #LPR on MAO-B
1-pchisq(7.06,4)  #LPR on MAO-A
1-pchisq(4.714,4)  #LPR on HT1A
1-pchisq(8.25,4)  #MAO-B on VNTR
1-pchisq(12.466,4)  #MAO-B on LPR
1-pchisq(9.669,4)  #MAO-B on MAO-A
1-pchisq(8.019,4)  #MAO-B on HT1A
1-pchisq(0.431,4)  #MAO-A on VNTR
1-pchisq(1.77,4)  #MAO-A on LPR
1-pchisq(3.494,4) #MAO-A on MAO-B
1-pchisq(2.791,4)  #MAO-B on HT1A
1-pchisq(3.166,4)  #HT1A on VNTR
1-pchisq(1.919,4)  #HT1A on LPR
1-pchisq(3.613,4)  #HT1A on MAO-B
1-pchisq(4.561,4)  #HT1A on MAO-A
```

Appendix V: S-Plus Code for Fisher's Exact Test and Three-way Interaction Test (ECT and Poisson)

```
dat<-read.table("c:\\matemp\\project\\data.txt",header=F)
#MAO-B
tbl<-table(dat$V5,dat$V2)
tbl<-tbl[,-c(1,2)]
fisher.test(tbl)

#LPR
tbl<-table(dat$V3,dat$V2)
tbl<-tbl[,-c(1,2)]
fisher.test(tbl)

#VNTR
tbl<-table(dat$V4,dat$V2)
tbl<-tbl[,-c(1,2)]
fisher.test(tbl)

#HT1A
tbl<-table(dat$V6,dat$V2)
tbl<-tbl[-1,-c(1,2)]
fisher.test(tbl)

#MAO-A
tbl<-table(dat$V7,dat$V2)
tbl<-tbl[-1,-c(1,2)]
fisher.test(tbl)

# three-way interaction test for MAO-B*LPR*Response by ECT method
tbl<-table(dat$V3,dat$V2,dat$V5)
tbl<-tbl[,-c(1,2),]
dim<-c('V2','V3','V5')
formula.f<-y~V2*V3*V5
reduce<-'V2*V3*V5'
ect(tbl,dim,formula.f,reduce)

# three-way interaction test for MAO-B*LPR*Response by ECT method
tbl<-table(dat$V7,dat$V2,dat$V5)
tbl<-tbl[-1,-c(1,2),]
dim<-c('V6','V3','V5')
formula.f<-y~V6*V3*V5
reduce<-'V6*V3*V5'
ect(tbl,dim,formula.f,reduce)

#Poisson regression in testing three-way interaction MAO-B*LPR*Response
tbl<-table(dat$V3,dat$V5,dat$V2)
tab2<-tbl[,,c(-1,-2)]
Rsp<-c(rep("N",4),rep("P",4),rep("R",4))
LPR<-rep(c('l','s'),6)
MAOB<-rep(c(rep('A',2),rep('G',2)),3)
Frq<-as.vector(tab2)
dat1<-data.frame(Rsp,LPR,MAOB,Frq)
x.glm0<-glm(Frq~LPR*MAOB +
   Rsp,family=poisson,data=dat1,control=glm.control(maxit=50))
```

```
x.glm1<-
   glm(Frq~LPR*MAOB*Rsp,family=poisson,data=dat1,control=glm.control(maxit=50))
x.glm2<-glm(Frq~LPR*MAOB + MAOB*Rsp +
   LPR*Rsp,family=poisson,data=dat1,control=glm.control(maxit=50))
x.glm3<-glm(Frq~LPR*MAOB +
   LPR*Rsp,family=poisson,data=dat1,control=glm.control(maxit=50))
anova(x.glm1,x.glm2,test='chisq')
step1<-step(x.glm0,list(lower=formula(x.glm0),upper=~.^2))
step1$anova
```

Appendix VI: Antidepressant Treatment Data

Table A. Frequency table of antidepressant response versus the combination of polymorphisms

LPR	VNTR	MAOB	HT1A	MAOA	Response Frequencies		
					N	P	R
1	10	A	C	4	4	9	7
s	10	A	C	4	0	0	1
1	12	A	C	4	4	11	6
s	12	A	C	4	0	3	4
1	10	G	C	4	1	3	5
s	10	G	C	4	0	2	0
1	12	G	C	4	0	0	3
s	12	G	C	4	0	0	1
1	10	A	G	4	2	4	4
s	10	A	G	4	0	0	0
1	12	A	G	4	1	0	1
s	12	A	G	4	0	1	2
1	10	G	G	4	0	2	2
s	10	G	G	4	0	0	0
1	12	G	G	4	0	0	2
s	12	G	G	4	0	2	1
1	10	A	C	3	1	5	1
s	10	A	C	3	0	0	1
1	12	A	C	3	0	2	0
s	12	A	C	3	0	1	0
1	10	G	C	3	0	0	1
s	10	G	C	3	0	0	0
1	12	G	C	3	0	0	1
s	12	G	C	3	0	0	0
1	10	A	G	3	0	1	1
s	10	A	G	3	0	0	0
1	12	A	G	3	1	0	0
s	12	A	G	3	0	0	2
1	10	G	G	3	0	0	1
s	10	G	G	3	0	0	0
1	12	G	G	3	0	0	0
s	12	G	G	3	0	0	1
s	12	A	*	4	0	0	1

LPR: a 44 bp insertion/deletion polymorphism in the 5-HTT promoter region;

VNTR: a 17 bp VNTR within intron 2 of the 5-HTT gene;

HT1A: a C to G SNP in the serotonin receptor 5-HT1A promoter region

MAOA: a 30 bp VNTR within the MAO-A promoter; MAOB: an A to G SNP in intron 13 of MAO-B;

N: no response P: partial response R: response *: Missed value