NTNU
Norwegian University of
Science and Technology

A statistical simulation-based framework for sample size considerations in case-control SNP association studies

Erik Edsberg

Master of Science in Physics and Mathematics
Submission date: June 2008
Supervisor: Mette Langaas, MATH
Co-supervisor: Endre Anderssen, IKM/DMF

Norwegian University of Science and Technology
Department of Mathematical Sciences

Problem Description

The aim of this thesis work is to build a statistical simulation-based framework to be used prior to performing a Single Nucleotide
Polymorphism (SNP) case-control associaton study.
The primary objective of the framework is in assessing the effect of sample size.
We focus on a disease model based on a single biallelic SNP.

Assignment given: 21. January 2008
Supervisor: Mette Langaas, MATH

Preface

This is my master thesis at the Industrial Mathematics programme of NTNU Norwegian University of Nature Science and Technology. I would like to thank Associate Professor Mette Langaas for her guidance and encouragement. I would also like to thank Endre Anderssen and Maiken Bratt Elvestad for providing initial information required for the framework and a user account needed for doing the simulations. Without them the thesis work could not have been initiated.

Trondheim, June 22, 2008

Erik Edsberg

Abstract

In this thesis work, a statistical simulation-based framework is presented that is capable of making sample size considerations prior to case-control association studies with biallelic single-SNP disease models. The numbers of cases and controls to be simulated are specified by the user, along with the allele frequency of the disease SNP and the penetrances of its genotypes. Based on this, genotypes for the disease SNP as well as a number of other SNPs are simulated for cases and controls using the genomeSIM package, and disease status is assigned, producing a case-control data set. In an example run of the framework, the MAX test is applied and power performance in detecting the disease SNP is assessed for various sample sizes, demonstrating how the framework can be used for finding an appropriate sample size.

Multiple-SNP disease models are included in this thesis work for conceptual overview, but are not implemented in the framework. Direct association methods are used and not indirect association methods that utilise linkage disequilibrium. The scope of the thesis is to provide the framework, which is shown to work and is a first step towards more realistic analyses. With its modular structure, key modules such as the simulation package and the statistical association method can be easily replaced, and additional functionality such as multiple testing adjustments can be added at a later time for a more advanced tool.

Contents

Chapter 1

Introduction

Single nucleotide polymorphisms (SNPs, pronunciation: "snip") are sites in genomic DNA which are subject to single nucleotide pair variation between individuals. The site is called a locus, and the genotype at the locus is called an allele of the SNP. Since human chromosomes come in pairs, an individual can have either zero, one or two copies of any given SNP allele. Being sites of genomic variation, SNPs are interesting both as a physiologic phenomenon but also because of their implications on susceptibility for complex hereditary diseases such as breast cancer [Campbell and Heyer, 2007, 193] and multiple sclerosis [Hafler et al., 2007]. One has estimated the number of SNPs in the human genome to 3 million [Primrose and Twyman, 2003]. The task of detecting SNPs that are associated with susceptibility for diseases may require large-scale statistical methods such as screening and genome-wide association scans [Hirschhorn and Daly, 2005], or investigation of SNPs in narrower regions of the genome suggested by biologists or biomedical scientists in a candidate gene approach. The methods may be divided into those that use family-based (pedigree) data, including linkage analysis [Balding et al., 2003, pg. 893-], and those that use population-based case-control data [Balding, 2006]. Case-control studies are known from non-genetic epidemiology. The methodology of such studies were first studied in the 1950s [Breslow and Day, 1980]. They can however be applied to genetic epidemiology as well [Cordell and Clayton, 2005].

A case-control association data set may contain recorded status of a binary disease trait and recordings of genotypes at a large number of SNP loci for each individual in a sample. The goal of the case-control association study is to detect SNPs whose harmful genotypes are associated with positive disease status [Balding, 2006]. This can be done by assuming a biologically plausible statistical model for how the SNP genotypes in the data affect susceptibility for the disease, deriving a statistic and then, for each SNP (or SNPs, if multiple-SNP disease models are considered) in question, examining the statistic's value according to its dis-

tribution under the null hypothesis of H_0: no association. Improbable values of the statistic, and hence small p-values, indicate detected associated SNPs for the assumed model. In cases with a large numbers of tests, multiple testing corrections could be applied in order to adjust the number of false positive SNPs detected due to chance [Balding, 2006].

The process of genotyping a large number of SNPs for multiple individuals during data set construction contributes to the cost of a case-control association study, as does time spent by the researcher on analysis after the data set is obtained. With a given statistical model, analysis may fail to detect a number of associated SNPs, for instance due to a too small sample size being used [Hattersley and McCarthy, 2005]. Costs from failed studies can be reduced if, in advance of decisions to order data sets and initiate study, power considerations are made in order to weed out studies that are not likely to succeed. Power considerations can be made analytically or, if the statistical method is too complex, estimated by simulation.

Following this introduction, Chapter 2 contains biological background useful for the later disease models and statistical methods. Chapter 3 reviews notation and methods that can be used for single-SNP and multiple-SNP case-control association, as well as multiple testing, sample size and statistical power. Chapter 4 introduces a statistical simulation-based framework for power considerations, starting with a description of the simulation algorithm of the genomeSIM package, followed by a structural presentation of the framework and a suggested application including a conceptual test run. The results of the test run are presented in Chapter 5 and discussed along with the framework's potential for immediate and future results in Chapter 6, which concludes the thesis. The R-code of the framework is given in the appendix.

Chapter 2

Biological background

The human genome consists of the 23 chromosome pairs, of which 22 are autosomal pairs and one is the sex chromosome pair. Scattered among genes and intergenic DNA, SNPs are found all throughout the genome. In this thesis the following definitions will be used:

- *SNP*: A site (locus) on a chromosome whose allele is subject to significant variation across the human population (i.e. the most common allele is present in no more than 95% or 99% of the population, depending on the definition used [Lesk, 2007], [Campbell and Heyer, 2007]).

- *locus*: A clearly defined DNA site in the genome. Genes are located at loci, as are SNPs.

- *allele*: The genotype at a locus is called its allele. An allele of a locus can be analogous to a realisation of a random variable in statistics or to the value of a variable in a computer program. Here, it is assumed that the allele of a SNP is a nucleotide pair (A-T, T-A, G-C or C-G), which is denoted by a single letter (e.g. an M allele).

- *biallelic SNP*: A SNP whose allele can be one of two possibilities (e.g. M or N). Any other remaining possibilities are seldom or never found.

- *two-chromosome vs. one-chromosome SNP genotype*: On autosomal chromosome pairs, every SNP is present on both chromosomes. The two-chromosome SNP genotype is the resulting allele pair, e.g. NM (heterozygous genotype) or NN or MM (homozygous genotypes) for a biallelic SNP. This is contrasted to the one-chromosome SNP genotype, which for biallelic SNPs is single a M or N.

- *risk allele*: A SNP allele which is associated with increased risk for some disease.

- *causal/noncausal SNP*: A causal SNP is a SNP whose risk allele(s) is causing a disease (rather than for instance being correlated with it due to linkage disequilibrium (LD) with another causal SNP.) Noncausal SNPs do not cause disease.

- *phased/unphased genotypes*: 2-chromosome genotypes for which the haplotype phase is known, i.e. for each allele pair it is known which chromosomes the alleles reside on. Expression (2.1) shows the difference between phased and unphased genotypes, where i.s.o. means 'in some orientation':

$$\begin{bmatrix} N_1N_1 \\ M_2N_2 \\ N_3M_3 \\ N_4N_4 \end{bmatrix} \text{ (phased)}, \quad \begin{bmatrix} \text{Two } N_1 \text{ alleles} \\ \text{One } N_2\text{, one } M_2 \text{ allele i.s.o.} \\ \text{One } N_3\text{, one } M_3 \text{ allele i.s.o.} \\ \text{Two } N_4 \text{ alleles} \end{bmatrix} \text{ (unphased)} \qquad (2.1)$$

- *haplotype*: A combined one-chromosome genotype of two or more SNPs (on a single chromosome). Combined two-chromosome genotypes of multiple SNPs consist of two haplotypes, of which one originates from the mother and the other from the father. In the phased genotype in (2.1), the two haplotypes are $N_1M_2N_3N_4$ and $N_1N_2M_3N_4$, respectively.

- *allele frequency*: The frequency with which a given allele is found at a the locus of its SNP

- *Hardy-Weinberg equilibrium*: The Hardy-Weinberg principle states that under a set of idealised population conditions [Hartl and Jones, 2006], allele frequencies remain constant from generation to generation, which is referred to as the Hardy-Weinberg equilibrium.

In the remainder of this thesis, the genotype of a SNP is treated as its two-chromosome SNP genotype.

Chapter 3

Statistical theory

3.1 Notation: Single-SNP analyses

Consider a biallelic SNP with two possible alleles denoted M and N respectively. This allows for four possible phased genotypes for that SNP at the two chromosomes: $(N,\ N)$, $(N,\ M)$, $(M,\ N)$ and $(M,\ M)$. Assume that M is the allele that is suspected to cause increased disease risk. If we assume that the allele causes risk independently of which chromosome it resides on, then we can treat the two phased heterozygous genotypes as one, leaving $(N,\ N)$, $(N,\ M)$ and $(M,\ M)$ as the three possible unphased genotypes. A hypothetical unphased case-control data set for associations between a disease and a biallelic SNP is portrayed in Table 3.1.

In order to analyse such a data set the following notation can be adapted from Zheng and Gastwirth [2006]. Notation of counts of cases and controls by genotype can be found in Table 3.2.

Here, the index i in the number of cases r_i, controls s_i, and total n_i, $i = 0, 1, 2$, equals the number of copies of allele M in the corresponding genotype. Next, notation for risk of disease by genotype can be summarized as in Table 3.3.

Here, (p_0, p_1, p_2), (p_0, p_1, p_2), and (q_0, q_1, q_2) are the frequencies of genotypes

Disease status	0	1	1	0	1	0	0	0	1	0
No. of M alleles	2	1	1	1	2	1	0	1	2	0

Table 3.1: A hypothetical case-control association data set with one SNP and 10 individuals. For each individual we know if the individual is a case or a control. Row 1 indicates case/control and row 2 indicates the number of copies of allele M.

	NN	NM	MM	Total
Case	r_0	r_1	r_2	r
Control	s_0	s_1	s_2	s
Total	n_0	n_1	n_2	n

Table 3.2: Counts of cases and controls by genotype

Symbol	Description
(g_0, g_1, g_2)	Probability of genotype (NN, NM and MM) in population
(p_0, p_1, p_2)	Probability of genotype (NN, NM and MM) in cases
(q_0, q_1, q_2)	Probability of genotype (NN, NM and MM) in controls
(f_0, f_1, f_2)	Penetrance, $P(\text{case} \mid \text{genotype})$
$K = \sum_{i=0}^{2} g_i f_i$	Prevalence of disease, $P(\text{case})$
g	Frequency of allele M

Table 3.3: Notation for risk of disease by genotype

NN, NM, and MM in the population, in cases, and in controls, respectively. Further, (f_0, f_1, f_2) are the penetrances: $f_i = P(\text{case} \mid i \text{ copies of allele } M \text{ in SNP genotype})$. The prevalence of disease in the population is $K = P(\text{case})$.

Using the theorem of total probability, $K = \sum_{i=0}^{2} g_i f_i$. It can be found from the definition of conditional probabilities that

$$\begin{aligned} p_i =& P(i \text{ copies of } M \mid \text{case}) = \frac{P(\text{case} \mid i \text{ copies of } M) P(i \text{ copies } of M)}{P(\text{case})} \\ =& \frac{f_i g_i}{K}, \ i = 0, 1, 2 \end{aligned}$$

Similarly,

$$q_i = \frac{(1 - f_i) g_i}{1 - K}$$

Let g be the allele frequency of M, i.e. the probability that a given allele of the SNP in question is an M. If the Hardy-Weinberg equilibrium holds, the relationship between (g_0, g_1, g_2) and g is

$$g_0 = (1 - g)^2, \ g_1 = 2(1 - g)g, \ g_2 = g^2 \tag{3.1}$$

In fact, the hypothetical data set displayed in Table 3.1 was simulated in R using a value of $g = 0.5$, then using (3.1) to calculate $(g_0, g_1, g_2) = (0.25, 0.5, 0.25)$ which was used to simulate the number of M alleles for each individual, and then

finally assuming values for penetrances, $(f_0, f_1, f_2) = (0.1, 0.5, 0.8)$, to determine disease status.

Genetic models for how phenotypes are expressed based on genotypes influence the penetrances. If the simplistic assumption is made that there is just one SNP and that the disease risk depends only on the genetic component of that SNP, then there are several ways to model that genetic dependence statistically (see Figure 3.1):

- The additive model, in which each additional M allele in the genotype increases the disease risk, resulting in a penetrance relationship which increases in some incremental pattern of $f_0 < f_1 < f_2$.
- The dominant model, in which the one M allele in the heterozygous phenotype, MN, is sufficient to cause an effect similar to the two M alleles in the homozygous MM genotype, resulting in a penetrance relationship tendency towards $f_0 < f_1 \approx f_2$. This suggests a natural dichotomization of the NM and MM genotypes versus the NN genotype.
- The recessive model, in which the penetrance relationship tendency is $f_0 \approx f_1 < f_2$.
- The overdominant model, in which the heterozygous genotype has the largest effect on disease risk ($f_1 > f_0, f_2$).

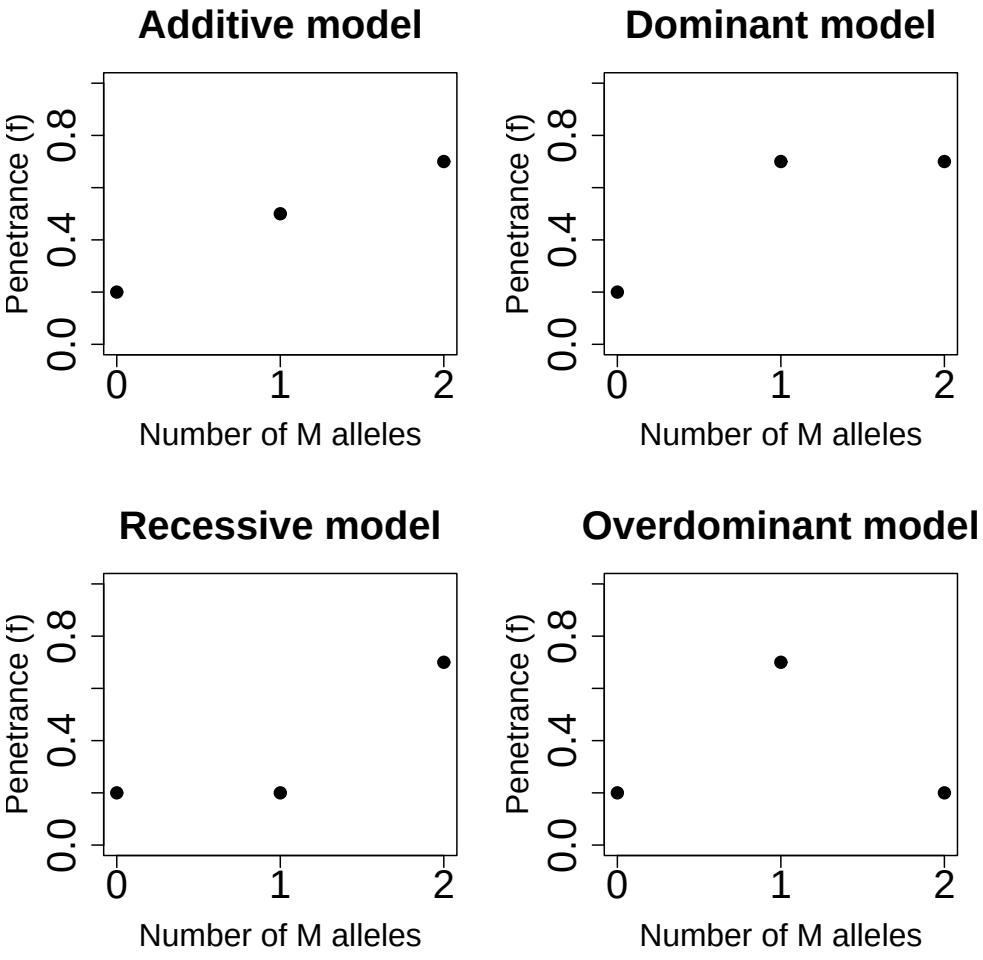

Figure 3.1: Genetic models that influence penetrances, illustrated using hypothetical penetrance values. From the upper left: Additive, dominant, recessive and overdominant model.

Model	**Term for $x^T\beta$**
intercept only	β_0
additive effect of allele	$\beta_0 + \beta_1 x_1$, $x_1 \in \{-1, 0, 1\}$
additive effect of allele, dominance	$\beta_0 + \beta_1 x_1 + \beta_2 x_2$, $x_1 \in \{-1, 0, 1\}$, $x_2 \in \{-0.5, 0.5, -0.5\}$
general	$I_{NN}\beta_0 + I_{NM}\beta_1 + I_{MM}\beta_2$
additive effect of allele	$(I_{NN} + \frac{1}{2}I_{NM})\beta_0 + (\frac{1}{2}I_{NM} + I_{MM})\beta_2$
dominance	$I_{NN}\beta_0 + (I_{NM} + I_{MM})\beta_2$

Table 3.4: Examples of models for unphased single-SNP logistic regression

3.2 Single-SNP direct association methods

Here, the application of two classes of statistical case-control methods for single-SNP direct association will be discussed. Indirect association is also very important in genetic epidemiology because most SNPs are not causal [Palmer and Cardon, 2005], but the theory of linkage disequilibrium that indirect association methods rest upon is beyond the scope of this thesis. However, the computer framework that is the focus of this thesis, and that will be described in a later section, has a modular structure that allows for indirect methods functionality to be included at a later time if needed.

The first of the two direct association methods that is described here is logistic regression. This will motivate the concept of odds ratios. Then, a class of contingency-table-based methods that include Pearson's test for independence, Cochran-Armitage's test for trend, and the MAX test, will be described.

3.2.1 Single-SNP logistic regression

Logistic regression can be used for modelling the relationship between the probability of an individual to obtain disease and the individual's SNP genotype [Balding, 2006]. The logistic regression model applied is

$$\text{logit}(\pi) = \ln\left(\frac{\pi}{1-\pi}\right) = x^T\beta \tag{3.2}$$

where the left-hand term is denoted as the 'log odds' of obtaining disease, π is the prospective[1] probability that the individual will obtain the disease given the

[1]Since π is assumed to be a prospective random response variable and the covariates x are

genotype, and $x^T\beta$ defines the genetic model through which the SNP genotype is assumed to affect disease risk.

Since the quantity π to which we want to fit a regression model is a probability, it is desirable to make sure that $0 \leq \pi \leq 1$. The logit transform in (3.2) ensures this property [Dobson, 2002, pg. 116,118].

The choice of $x^T\beta$ defines the genetic model used. Here, two approaches for application of logistic regression in unphased single-SNP case-control association will be considered. Cordell and Clayton [2002] define three single-SNP models as a part of their larger generalised linear model (GLM) multiple-SNP logistic regression framework. Balding [2006] provides three different single-SNP models. The six models are listed in Table 3.4. We start by looking at the simplest model,

$$\begin{aligned} &x^T\beta = \beta_1 \\ &x = 1,\ \beta = \beta_1 \end{aligned} \tag{3.3}$$

This model features only an intercept term and does not take into account genotypes. The next model, given unphased genotypes $(N,\ N)$, $(N,\ M)$ and $(M,\ M)$, takes into account the additive effect of allele M by coding the genotypes as -1, 0 and 1, respectively, and adding a parameter β_1, resulting in the model

$$\begin{aligned} &x^T\beta = \beta_1 x_1 + \beta_2 x_2,\ x_1 = 1,\ x_2 \in \{-1, 0, 1\} \\ &x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in \left\{ \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right\},\ \beta = \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix} \end{aligned} \tag{3.4}$$

This models an additive allele effect because each copy of allele M in the genotype adds a value β_2 to the log odds. If in addition it is desirable to model genetic dominance of allele M over N, then the effect of the heterozygous genotype must be increased. Another parameter β_3 with an appropriately coded covariate, for instance $x_3 \in \{-0.5, 0.5, -0.5\}$, achieves this, resulting in a dominant model,

assumed to be fixed, the logistic regression model for disease probabilities is well suited for cohort studies [Breslow and Day, 1980]. But it is important to be aware that case-control studies are often retrospective rather than prospective, i.e. the cases and controls are selected in retrospect according to disease status. The possible equivalence [Balding, 2006] of prospective and retrospective analysis in genetic edidemiological logistic regression is beyond the scope of this thesis.

$$
\begin{aligned}
&x^T\beta = \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3,\ x_1 = 1,\ x_2 \in \{-1, 0, 1\},\ x_3 \in \{-0.5, 0.5, -0.5\} \\
&x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \in \left\{ \begin{bmatrix} 1 \\ -1 \\ -0.5 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 0.5 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ -0.5 \end{bmatrix} \right\},\ \beta = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix}
\end{aligned} \tag{3.5}
$$

The first of the three models of Balding [2006] can be represented as follows:

$$
\begin{aligned}
&x^T\beta = I_{NN}\beta_1 + I_{NM}\beta_2 + I_{MM}\beta_3 \\
&x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \in \left\{ \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \right\},\ \beta = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix}
\end{aligned} \tag{3.6}
$$

where I are indicator functions taking values 1 if an individual has the corresponding genotype and 0 if not. Additive effects of allele M can then be modelled by setting $\beta_2 = \frac{1}{2}(\beta_1 + \beta_3)$, such that the heterozygous genotype's contribution to the log odds becomes the middle value of the two homozygous ones. Dominance can be achieved by setting $\beta_2 = \beta_3$, so that the two genotypes that contain the dominant allele contribute equally to disease risk. The additive and dominant models resulting from this modification of (3.6) are displayed in Table 3.4.

The logistic regression model (3.2) can be formulated in the generalized linear model framework (GLM), which enables maximum likelihood parameter estimation and hypothesis testing based on χ^2-distributed goodness-of-fit statistics, such as the log-likelihood ratio statistic, for comparisons of different nested models [Dobson, 2002, pg. 115-18]. For instance, (3.3) can be arrived at from (3.6) by demanding $\beta_1 = \beta_2 = \beta_3$, so models (3.3) and (3.6) are nested. The corresponding log-likelihood ratio statistic has as χ^2_2 distribution under the null hypothesis that both models fit the data well, so if the computed statistic value does not have an unlikely value, model (3.3) is preferred in place of (3.6) since it is the simpler model. This is logically equivalent to testing the hypothesis $H_0 :\ \beta_1 = \beta_2 = \beta_3$.

3.2.2 Methods based on contingency tables

The Cochran-Armitage (CA) test for trend was presented independently by Cochran [1954] and Armitage [1955]. Assume as a model that the numbers of cases r_i, $i = 0, 1, 2$, from the contingency table, Table 3.2, are independent binomial variables with

$$\begin{aligned} &R_0, ..., R_2 \text{ independent} \\ &R_i \sim \text{binom}(n_i, \pi_i) \end{aligned} \tag{3.7}$$

where π_i is the corresponding risk of obtaining disease for an individual in group i. Assume further that there is a linear trend in the π_is on the form

$$\pi_i = \alpha + \beta x_i \tag{3.8}$$

Why would we want to assume such a linear trend? This corresponds to assuming what Balding [2006] refers to as additive risk, meaning that multiple alleles of the risk SNP contribute in an additive sense to the susceptibility of disease. To quote Balding [2006]:

> 'For complex traits, it is widely thought that contributions to disease risk from individual SNPs will often be roughly additive'

Given the model defined in (3.7) and (3.8), then the CA test for trend is based [Agresti, 2002, pg. 181-2] on the hypothesis

$$H_0 : \ \beta = 0 \text{ vs. } H_1 : \ \beta \neq 0 \tag{3.9}$$

The corresponding CA statistic can be derived by obtaining weighted least squares-estimates for α and β in equation (3.8), then using these estimates in a partitioning of the Pearson goodness-of-fit χ^2-statistic for model (3.7) into two new χ^2-terms [Agresti, 2002, pg. 181-2]. One of these two terms is the CA statistic. Next, it will be explained what the numbers x_i are, how the least-squares estimates for α and β are obtained, what the Pearson goodness-of-fit statistic is for this model and its partitioning into the CA statistic will be listed.

The numbers x_i

If we assume (3.8), then we also assume a natural ordering of the π_is. If β is positive, then we would expect that $\pi_0 \leq \pi_1 \leq \pi_2$. In the application of the CA test for trend, the numbers x_i are to be chosen in order to reflect this ordering. There are many possible choices for x_i. For example, in our situation with 0, 1 or 2 copies of a risk allele of a given SNP, the following choices for (x_0, x_1, x_2) all reflect an additive model: $(0, 1, 2)$, $(1, 2, 3)$, and $(-1, 0, 1)$. Other choices such as $(0, 1, 1)$ and $(0, 0, 1)$ do not reflect a linear trend but instead, dominant and recessive models, respectively. These other types of models will be utilised by the MAX test at the end of this section.

Weighted least-squares estimates of α and β

In this passage, Agresti [2002, pg. 181-2] is used as a reference for investigating the CA test for trend observator. Some details will provided here that were omitted in that reference. The expressions obtained in Agresti [2002, pg. 181-2] for the weighted least-squares estimates a and b of α and β from (3.8), are[2]

$$\begin{aligned} a &= p - b\bar{x} \\ b &= \frac{\sum_{i=1}^{I} n_i(p_i - p)(x_i - \bar{x})}{\sum_{i=1}^{I} n_i(x_i - \bar{x})^2} \end{aligned} \tag{3.10}$$

where we use as a notation $p_i = \frac{y_i}{n_i}$, $p = \frac{1}{n}\sum_{i=1}^{I} y_i$, and $\bar{x} = \frac{1}{n}\sum_{i=1}^{I} n_i x_i$. It will now be shown how these expressions can be obtained. One way to obtain (3.10) is by starting with a weighted sum of squared errors with unspecified weights ω_i [Wood, 1978]:

$$\sum_{i=1}^{I} \omega_i(\pi_i - \alpha - \beta x_i)^2 \approx \sum_{i=1}^{I} \omega_i(p_i - \alpha - \beta x_i)^2 \tag{3.11}$$

In Agresti [2002, pg. 181-2], the weights used are $\omega_i = n_i$. This may be motivated by the following: Denote the binomial random variable for the number of cases in group i by Y_i. In (3.11), the random variable of which we are taking the sum of squares is $\frac{Y_i}{n_i}$. Its expected value is $\frac{n_i\pi_i}{n_i} = \pi_i$ and its variance is $\frac{1}{n_i^2} n_i\pi_i(1 - \pi_i) = \frac{\pi_i(1-\pi_i)}{n_i}$, which depends on n_i. Thus small n_is yield small terms in the sum (3.11). The choice of $\omega_i = n_i$ can be viewed as a counterweight of this.

Inserting $\omega_i = n_i$ into (3.11), the equations for minimizing the sum of squares are

[2]In (3.8), x_i ranges from 0 to 2, which should be substituted for i=1 to I in (3.8) for use in this thesis. This applies to all expressions in this section with sums ranging from i=1 to I.

$$
\begin{aligned}
0 &= \frac{\partial}{\partial\alpha}\sum_{i=1}^{I} n_i(p_i-\alpha-\beta x_i)^2 = -2\sum_{i=1}^{I} n_i(p_i-\alpha-\beta x_i) \\
&\Downarrow \\
\alpha\sum_{i=1}^{I} n_i &= \sum_{i=1}^{I} n_i(p_i-\beta x_i) \\
&\Downarrow \\
\alpha &= \frac{\sum_{i=1}^{I} n_i p_i}{\sum_{i=1}^{I} n_i} - \beta\frac{\sum_{i=1}^{I} n_i x_i}{\sum_{i=1}^{I} n_i} = p-\beta\bar{x}
\end{aligned}
$$

and

$$
0 = \frac{\partial}{\partial\beta}\sum_{i=1}^{I} n_i(p_i-\alpha-\beta x_i)^2 = -2\sum_{i=1}^{I} n_i x_i(p_i-\alpha-\beta x_i) \tag{3.12}
$$

Thus the least squares estimate of α becomes $a = p - b\bar{x}$. It remains to determine b, the least squares estimate of β. By inserting $\alpha = a = p - \beta\bar{x}$ into (3.12), the expression for b in (3.10) can be obtained in the following way:

$$
\begin{aligned}
0 &= -2\sum_{i=1}^{I} n_i x_i(p_i-p+\beta\bar{x}-\beta x_i) \\
&\Downarrow \\
\beta &= \frac{\sum_{i=1}^{I} n_i x_i(p_i-p)}{\sum_{i=1}^{I} n_i x_i(x_i-\bar{x})} \\
&= \frac{\sum_{i=1}^{I} n_i x_i p_i - p\sum_{i=1}^{I} n_i x_i + p\bar{x}n - p\bar{x}n}{\sum_{i=1}^{I} n_i x_i^2 - 2\bar{x}\sum_{i=1}^{I} n_i x_i + \bar{x}\sum_{i=1}^{I} n_i x_i} \\
&= \frac{\sum_{i=1}^{I} n_i x_i p_i - p\sum_{i=1}^{I} n_i x_i + p\bar{x}n - \bar{x}\sum_{i=1}^{I} n_i p_i}{\sum_{i=1}^{I} n_i x_i^2 - 2\bar{x}\sum_{i=1}^{I} n_i x_i + \bar{x}^2 n} \\
&= \frac{\sum_{i=1}^{I} n_i(p_i x_i - p_i\bar{x} - p x_i + p\bar{x})}{\sum_{i=1}^{I} n_i(x_i^2 - 2x_i\bar{x} + \bar{x}^2)} \\
&= \frac{\sum_{i=1}^{I}(p_i-p)(x_i-\bar{x})}{\sum_{i=1}^{I} n_i(x_i-\bar{x})^2} \equiv b
\end{aligned}
$$

Pearson Goodness-of-fit statistic for this model

As a part of the derivation of the CA statistic in Agresti [2002, pg. 181-2], the expression

$$\frac{1}{p(1-p)}\sum_{j=1}^{I} n_j(p_j - p)^2$$

is given as the 'Pearson statistic for testing independence'. Here, it will be explained how this expression can be obtained. Consider Table 3.2. Assuming model (3.7), the Pearson Goodness-of-fit [Walpole et al., 2002] for that model is

$$X^2(I) = \sum_{i=1}^{2}\sum_{j=1}^{I} \frac{(o_{ij} - e_{ij})^2}{e_{ij}} \sim \chi^2_{I-1}$$

where o_{ij} and e_{ij} are the observed and expected frequencies of the cell in the ith row and jth column of Table 3.2 under (3.7). The observed frequencies are

$$o_{ij} = \begin{cases} r_j, & i = 1 \\ n_j - r_j, & i = 2 \end{cases}$$

Under (3.7), the expected frequency of cell ij is $n \times P$(person is assigned to cell ij). If we define the events

$$\begin{aligned} A_{ij}&: \text{ Person is assigned to cell } ij \\ B_i&: \text{ Person is assigned to row } i \\ C_j&: \text{ Person is assigned to column } j \end{aligned}$$

then under (3.7),

$$\begin{aligned} P(A_{ij}) &= P(B_i \cap C_j) = P(B_i)P(C_j) \\ &= \begin{cases} \frac{\sum_{s=1}^{I} r_s}{n} \times \frac{n_j}{n} = \frac{n_j(\sum_{s=1}^{I} r_s)}{n^2}, & i = 1 \\ \frac{\sum_{s=1}^{I}(n_s - r_s)}{n} \times \frac{n_j}{n} = \frac{n_j \sum_{s=1}^{I}(n_s - r_s)}{n^2}, & i = 2 \end{cases} \end{aligned}$$

such that

$$e_{ij} = \begin{cases} \frac{n_j \sum_{s=1}^{I} r_s}{n}, & i = 1 \\ \frac{n_j \sum_{s=1}^{I}(n_s - r_s)}{n}, & i = 2 \end{cases}$$

Hence,

$$
\begin{aligned}
v^2(I) &= \sum_{i=1}^{2}\sum_{j=1}^{I}\frac{(o_{ij}-e_{ij})^2}{e_{ij}} \\
&= \sum_{j=1}^{I}\left[\frac{\left(r_j - \frac{n_j(\sum_{s=1}^{I} r_s)}{n}\right)^2}{\frac{n_j(\sum_{s=1}^{I} r_s)}{n}} + \frac{\left(n_j - r_j - \frac{n_j(\sum_{s=1}^{I}(n_s - r_s))}{n}\right)^2}{\frac{n_j(\sum_{s=1}^{I}(n_s - r_s))}{n}}\right]
\end{aligned} \tag{3.13}
$$

By inserting $r_i = n_i p_i$ and utilizing the notation $p = \frac{1}{n}\sum_{s=1}^{I} r_s$, this expression can be simplified in the following way:

$$
\begin{aligned}
v^2(I) &= \sum_{j=1}^{I}\left[\frac{\left(r_j - \frac{n_j(\sum_{s=1}^{I} r_s)}{n}\right)^2}{\frac{n_j(\sum_{s=1}^{I} r_s)}{n}} + \frac{\left(n_j - r_j - \frac{n_j(\sum_{s=1}^{I}(n_s - r_s))}{n}\right)^2}{\frac{n_j(\sum_{s=1}^{I}(n_s - r_s))}{n}}\right] \\
&= \sum_{j=1}^{I}\left[\frac{(r_j - n_j p)^2}{n_j p} + \frac{(n_j - r_j - n_j(1-p))^2}{n_j(1-p)}\right] \\
&= \sum_{j=1}^{I}\left[\frac{(1-p)(r_j^2 - 2r_j n_j p + n_j^2 p^2)}{p(1-p)n_j} + \frac{p(n_j p - r_j)^2}{p(1-p)n_j}\right] \\
&= \frac{1}{p(1-p)}\sum_{j=1}^{I}\left[\frac{r_j^2 - 2r_j n_j p + n_j^2 p^2}{n_j}\right] \\
&= \frac{1}{p(1-p)}\sum_{j=1}^{I}\left[\frac{n_j^2 p_j^2 - 2n_j^2 p_j p + n_j^2 p^2}{n_j}\right] = \frac{1}{p(1-p)}\sum_{j=1}^{I} n_j(p_j - p)^2
\end{aligned} \tag{3.14}
$$

which is the same expression as in Agresti [2002, pg. 181-2].

Partitioning of Pearson Goodness-of-fit statistic into CA statistic

The quantity (3.14) can be partitioned as follows [Agresti, 2002, pg. 181-2]:

$$X^2 \equiv X^2(I) = z^2 + X^2(L)$$

where

$$z^2 = \frac{b^2}{p(1-p)} \sum_{i=1}^{I} n_i (x_i - \bar{x})^2 = \left[\frac{\sum_{i=1}^{I} (x_i - \bar{x}) r_i}{\sqrt{p(1-p) \sum_{i=1}^{I} n_i (x_i - \bar{x})^2}} \right]^2$$

$$X^2(L) = \frac{1}{p(1-p)} \sum_{i=1}^{I} n_i (p_i - \hat{\pi}_i)^2$$

$$\hat{\pi}_i = p + b(x_i - \bar{x})$$

The CA statistic for testing $H_0 : \beta = 0$ in 3.8 is z^2, and it is asymptotically chi-square-distributed on 1 degree of freedom [Armitage, 1955].

The MAX test

One procedure which is related to the CA test for trend is the MAX test [Freidlin et al., 2002]. The MAX test aims at considering three of the four genetic models that were presented in Figure 3.1: The additive, recessive and the dominant model. These models can be reflected in a choice of weights in the CA test for trend. For instance, $\{-1, 0, 1\}$, $\{0, 0, 1\}$ and $\{0, 1, 1\}$ are three possible choices that correspond to the additive, recessive and the dominant model, respectively. These choices define three distinct CA tests, and it is true in every situation that at least one will be the most powerful of the three, depending on which genetic model is the correct one in that situation. However, the true genetic model is often unknown, so if the CA test for trend is used with one particular set of weights there is a risk that the weights corresponding to the true genetic model are not chosen. The consequential loss of power is addressed by Freidlin et al. [2002]. This motivates consideration of the MAX test because it is robust to the underlying true genetic model.

Denoting by Z_{ADD}, Z_{REC} and Z_{DOM} the CA statistics with weights corresponding to the three genetic models, the MAX test statistic is the maximum of the three statistics,

$$MAX = \max(Z_{ADD}, Z_{REC}, Z_{DOM}) \tag{3.15}$$

However, the asymptotic distribution of the maximum of the three CA χ^2-distributed test statistics is a difficult matter and the distribution is not known. However, the power the MAX test can be investigated by simulation [Freidlin et al., 2002].

In a master thesis which is related to the work of this thesis, the MAX test has been explored in the context of SNPs and multiple testing [Risberg, 2008]. In the remainder of this thesis, an R implementation will be used that is a slight modification of the MAX test, called *maxstat* [Gonzalez et al., 2008].

3.3 Odds ratios

The odds ratio is a common measure of effect size in genetic association studies. In Section 4.3, odds ratios will be used for obtaining penetrance values for simulations.

In a case-control setting, assume that association between disease and one risk factor is studied in a sample of individuals which is divided into two groups according to exposure of the risk factor. The first group contains individuals (including both cases and controls) exposed to the risk factor and the second group contains unexposed individuals (also cases and controls). For an individual, define $P_1 = P(\text{obtaining disease} \mid \text{exposed})$ and $P_2 = P(\text{obtaining disease} \mid \text{unexposed})$. Then the odds ratio ψ can be written as follows:

$$\psi = \frac{\frac{P_1}{1-P_1}}{\frac{P_2}{1-P_2}} \tag{3.16}$$

where the quantities $\frac{P_1}{1-P_1}$ and $\frac{P_2}{1-P_2}$ are known as the 'odds of disease' in the exposed and unexposed groups, respectively, hence the name 'odds ratio'.

When one biallelic SNP is used as a risk factor, then the quantities P_1 and P_2 in (3.16) are related to the penetrances f_0, f_1 and f_2. In order to express the odds ratio in terms of penetrances one must dichotomise the data by defining which of the genotypes, NN, NM and MM, that correspond to the exposed group and which that correspond to the unexposed group. For an odds ratio of the MM genotype (exposed) versus the NN genotype (unexposed) for instance, $P_1 = f_2$ and $P_2 = f_0$, so the odds ratio is

$$\psi = \frac{\frac{f_2}{1-f_2}}{\frac{f_0}{1-f_0}} \tag{3.17}$$

If the genotypes are counted according to Table 3.2, then an estimate of a penetrance f_i is $\hat{f}_i = \frac{r_i}{n_i}$, so an estimate of the odds ratio of the MM genotype versus NN can be calculated as

$$\hat{\psi}_{MM,NN} = \frac{\frac{\frac{r_2}{n_2}}{1-\frac{r_2}{n_2}}}{\frac{\frac{r_0}{n_0}}{1-\frac{r_0}{n_0}}} \tag{3.18}$$

Similarly, one could compute the odds ratios of MN versus NN or MM versus NM. If the dominant model is assumed, one could dichotomise by combining the MM and NM genotypes as the exposed group and calculating their odds ratio versus NN as the unexposed group. The odds ratio for the dominant model can be

derived as follows: Since NN is the unexposed group, $P_2 = f_0$ as before. Denote by D the event that an individual obtains disease, and denote by A and B the events that an individual has the genotypes NM and MM, respectively. Then,

$$P_1 = P(D \mid A \cup B) = \frac{P(D \cap (A \cup B))}{P(A \cup B)} = \frac{P(D \cap (A \cup B))}{P(A) + P(B)} \tag{3.19}$$

since A and B are disjoint. Further, by applying DeMorgan's law,

$$\begin{aligned} P(D \cap (A \cup B)) &= P((D \cap A) \cup (D \cap B)) \\ &= P(D \cap A) + P(D \cap B) - 0 \\ &= P(D \mid A)P(A) + P(D \mid B)P(B) \end{aligned} \tag{3.20}$$

So

$$P_1 = \frac{P(D \mid A)P(A) + P(D \mid B)P(B)}{P(A) + P(B)} = \frac{f_1 g_1 + f_2 g_2}{g1 + g2} \tag{3.21}$$

and the odds ratio is

$$\psi_{MM+NM,NN} = \frac{\frac{\frac{f_1 g_1 + f_2 g_2}{g1+g2}}{1 - \frac{f_1 g_1 + f_2 g_2}{g1+g2}}}{\frac{f_0}{1-f_0}} \tag{3.22}$$

Inserting estimates $\hat{f}_i = \frac{r_i}{n_i}$ and $\hat{g}_i = \frac{n_i}{n}$, an estimate of (3.22) is

$$\hat{\psi}_{MM+NM,NN} = \frac{\frac{\frac{r_2+r_1}{n_2+n_1}}{1 - \frac{r_2+r_1}{n_2+n_1}}}{\frac{\frac{r_0}{n_0}}{1-\frac{r_0}{n_0}}} \tag{3.23}$$

There is a connection between the odds ratio and the logistic regression model (3.2), because the left-hand side of Equation (3.2) is the log of the odds that appear in the odds ratio. The odds ratio is also related to the quantity known as the relative risk. The picture can be outlined as follows:

$$\text{logistic regression} \leftrightarrow \text{odds} \leftrightarrow \text{odds ratio} \leftrightarrow \text{relative risk}$$

Breslow and Day [1980, pg. 57,70-71] describes the relationship between odds ratio and the relative risk. Given the situation with one disease and one risk factor, the relative risk r is

$$r = \frac{P_1}{P_2} \tag{3.24}$$

that is, the ratio of incidence of exposed individuals versus unexposed individuals. When the probabilities P_1 and P_0 are small, the odds ratio (3.16) can be used for approximating the relative risk (3.24) due to the following argument [Breslow and Day, 1980]:

$$\begin{aligned} P_1,\ P_0 \text{ small} &\Rightarrow 1 - P_1 \approx 1 - P_0 \approx 1 \\ &\Rightarrow \psi = \frac{\frac{P_1}{1-P_1}}{\frac{P_0}{1-P_0}} \approx \frac{P_1}{P_0} = r \end{aligned}$$

Thus in the situation of one SNP with genotypes NN, NM and MM, the corresponding odds ratio-estimates of the relative disease risk of MM versus NN is expression (3.18). For the dominant model, the estimate of the relative risk of dominant genotypes MM and NM versus NN is expression (3.23).

A convenient property of this estimate of the relative risk using the odds ratio is that it is valid [Breslow and Day, 1980, pg. 71] both for prospective studies and for retrospective studies such as the case-control studies that are the focus of this thesis.

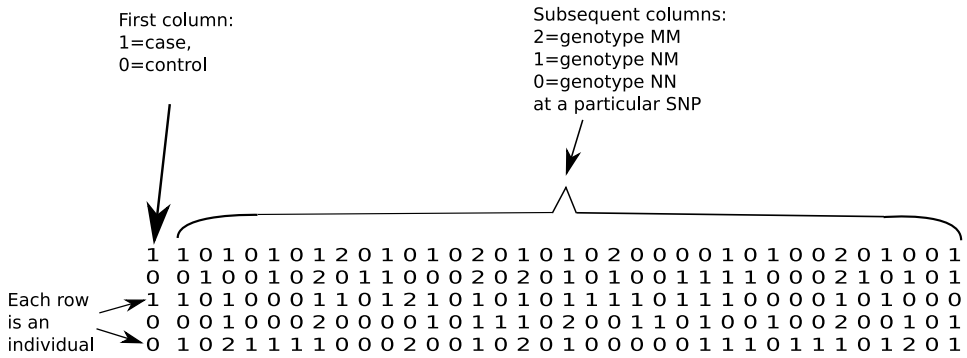

Figure 3.2: A multiple-SNP case-control data set, created by genomeSIM. Each row corresponds to one individual. The first column indicates disease status and the subsequent columns indicates the genotype, i.e. the number of copies of M alleles, at each SNP.

3.4 Notation: Multiple-SNP analyses

Previously in this section, notation and case-control methods for detecting associations between disease and one SNP was discussed. The notation for multi-locus data is similar to that of the one-SNP case in Table 3.1 . For multiple SNPs, Table 3.1 can be expanded into that shown in Figure 3.2, which shows a multiple-SNP data set created by the genomeSIM genomic simulation package. See Section 4.1 for details on the genomeSIM package. The difference from Table 3.1 is that there are additional columns corresponding to additional SNPs.

Assuming direct association, the step from single causal SNPs to multiple causal SNPs increase the genetic model options that can be incorporated into the multiple-SNP statistical methods. Some of these additional genetic model options will now be discussed.

Parent-of-origin, haplotype, main and epistasis effects

Consider two biallelic SNP loci on a chromosome. Let the possible alleles of the first alleles be $\{M_1, N_1\}$ and $\{M_2, N_2\}$, respectively. In the following, the approach of Cordell and Clayton [2002] will be used to explain the concepts of parent-of-origin effects, haplotype effects, main effects and epistasis effects in terms of genotypes

$\begin{bmatrix} N_1N_1 \\ N_2N_2 \end{bmatrix}$	$\begin{bmatrix} N_1N_1 \\ N_2M_2 \end{bmatrix}$	$\begin{bmatrix} N_1M_1 \\ N_2N_2 \end{bmatrix}$	$\begin{bmatrix} N_1M_1 \\ N_2M_2 \end{bmatrix}$
$\begin{bmatrix} N_1N_1 \\ M_2N_2 \end{bmatrix}$	$\begin{bmatrix} N_1N_1 \\ M_2M_2 \end{bmatrix}$	$\begin{bmatrix} N_1M_1 \\ M_2N_2 \end{bmatrix}$	$\begin{bmatrix} N_1M_1 \\ M_2M_2 \end{bmatrix}$
$\begin{bmatrix} M_1N_1 \\ N_2N_2 \end{bmatrix}$	$\begin{bmatrix} M_1N_1 \\ N_2M_2 \end{bmatrix}$	$\begin{bmatrix} M_1M_1 \\ N_2N_2 \end{bmatrix}$	$\begin{bmatrix} M_1M_1 \\ N_2M_2 \end{bmatrix}$
$\begin{bmatrix} M_1N_1 \\ M_2N_2 \end{bmatrix}$	$\begin{bmatrix} M_1N_1 \\ M_2M_2 \end{bmatrix}$	$\begin{bmatrix} M_1M_1 \\ M_2N_2 \end{bmatrix}$	$\begin{bmatrix} M_1M_1 \\ M_2M_2 \end{bmatrix}$

Table 3.5: Complete set of genotypes for 2 biallelic SNPs

for a situation of two bialellic SNPs. For an individual, the possible genotypes of the two SNPs on the two chromosomes is displayed in Table 3.5.

From the definition of haplotypes in Chapter 2, we know that each genotype in Table 3.5 consists of two haplotypes of which one originates from each parent. A possible assumption one can make is no parent-of-origin effects, which means that a haplotype has equal effect on disease risk independently of which parent it originated from. Under this assumption, the following two genotypes are assumed to have an equal effect on disease risk, which means that they can be treated as one:

$$\begin{bmatrix} N_1N_1 \\ N_2M_2 \end{bmatrix} \text{ and } \begin{bmatrix} N_1N_1 \\ M_2N_2 \end{bmatrix}$$

Merging all genotypes that have equal effects under the assumption of no parent-of-origin effects, Table 3.5 reduces to Table 3.6.

If in addition one is willing to assume that there are no haplotype effects, then one additional genotype can be removed from Table 3.6. Consider the two genotypes

$$\begin{bmatrix} N_1M_1 \\ N_2M_2 \end{bmatrix} \text{ and } \begin{bmatrix} N_1M_1 \\ M_2N_2 \end{bmatrix}$$

These genotypes contain the same number of N and M alleles on both SNPs, but they have different haplotypes. If there was a haplotype effect, then one could suspect that the genotypes had different disease risks. Otherwise, only the presence of alleles would count and one would assume the risks to be equal. Merging the two genotypes, Table 3.6 reduces to Table 3.7.

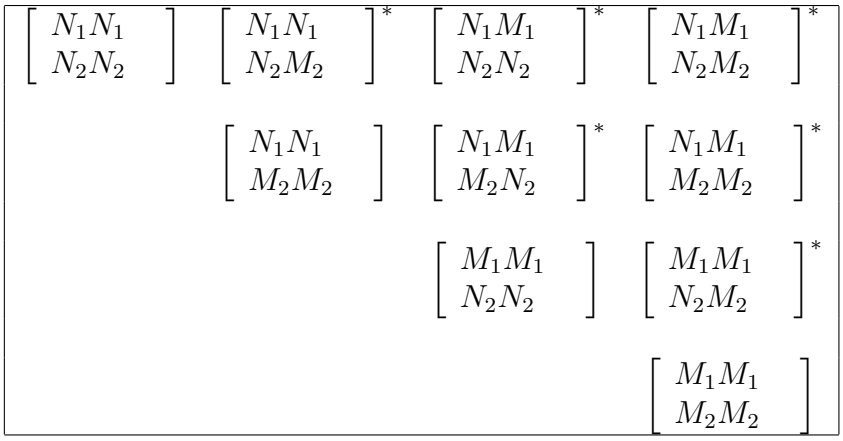

Table 3.6: Set of genotypes for 2 biallelic SNPs, under assumption of no parent-of-origin effects. * indicates that the genotype is merged with and contains another genotype.

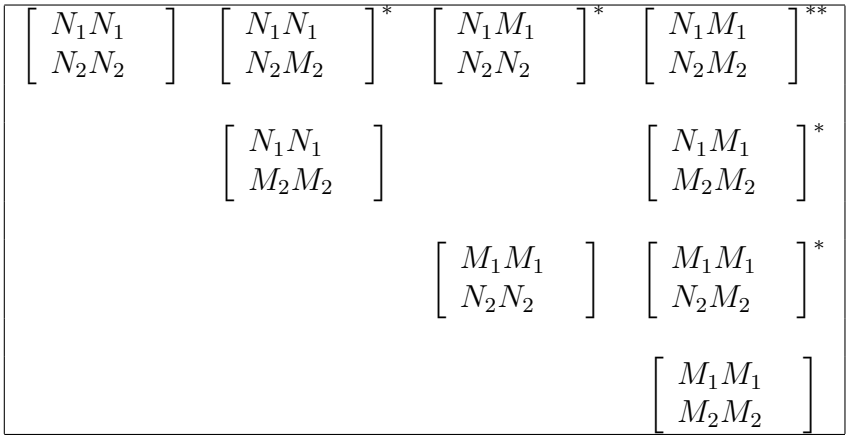

Table 3.7: Set of genotypes for 2 biallelic SNPs, under assumption of no parent-of-origin effects and no haplotype effects. *(**) indicates that the genotype is merged with and contains another (two) genotype(s).

The assumption of no haplotype effects has the benefit that one can use genotype data for which the phase is unknown, so-called unphased data. The assumption is reasonable [Cordell and Clayton, 2002] if one suspects that the SNPs in question are causal SNPs that are directly associated with the disease. If on the other hand the SNPs are noncausal and indirectly associated with the disease through (i.e. in linkage disequilibrium with) some other unknown causal SNP, the assumption can not be expected to hold. Since this thesis assumes direct association using unphased data, Table 3.7 will be used in the following, reducing the number of distinct 2-SNP genotypes from 16 to 9.

Main effects are effects on disease risk due to single loci [Hoh and Ott, 2003].

The last type of effect that will be considered here is that of epistasis, interaction between different SNPs. Though any detection of epistasis may reveal only little about the true underlying biological process, inclusion of epistasis in the statistical analysis can nevertheless improve power to detect genetic effects [Cordell, 2002]. In terms of the genotypes in Table 3.7, assume that the alleles M_1 and M_2 are the alleles that are assumed to carry increased risk for disease. Cordell and Clayton [2002] consider four epistasis effects due to the four 2-SNP genotypes in Table 3.7 that contain at least one risk allele of each type:

$$\begin{bmatrix} N_1M_1 \\ N_2M_2 \end{bmatrix}^{**}, \begin{bmatrix} N_1M_1 \\ M_2M_2 \end{bmatrix}^{*}, \begin{bmatrix} M_1M_1 \\ N_2M_2 \end{bmatrix}^{*}, \text{ and } \begin{bmatrix} M_1M_1 \\ M_2M_2 \end{bmatrix}$$

In this case, any increased epistasis effect on disease risk would be due to interaction between the two SNPs in these four 2-SNP risk genotypes. This is contrasted to a situation where only the main effects of the two SNPs are considered, in which no such interactions would be assumed to increase disease risk.

In the remainder of this thesis it is assumed that parent-of-origin effects and haplotype effects can be ignored. Hence, only the genotypes in Table 3.7 will be considered.

3.5 Multiple-SNP direct association methods

The previous subsections dealt with case-control methods for direct association between one SNP and a disease. Assuming a data set such as the one displayed in Figure 3.2, there might be more than one single SNP that is associated with the disease. In such a data set, one has the option to apply a single-SNP method to every single one of the m SNPs in the set. The result is a list of m p-values, some of which could be significantly small. But since there could be false positives due to chance, the issue of multiple testing has to be taken into account. Multiple testing will be handled in a later subsection. This procedure can be labelled as 'single-locus search' Hoh and Ott [2003] and it is capable of detecting a number of potentially causal SNPs. But since it considers only main effects, it can neglect information about joint effects of the SNPs [Balding, 2006]. Therefore, other methods have been developed that model the combined effect of multiple SNPs rather than multiple testing corrections of single-SNP search. These multi-locus methods and are still in an early phase of development [Cordell and Clayton, 2005].

Two possible adaptation of the logistic regression model to the setting of multiple-SNP case-control genetic epidemiology will now be explained.

3.5.1 Multiple-SNP logistic regression

Multiple-SNP logistic regression can be viewed as an expansion to the single-SNP logistic regression model that was discussed in Section 3.2. Like in Equation (3.2), π is still going to be the disease risk, i.e. the probability of obtaining disease given genotype, but the right-hand term, $x^T\beta$, will change to incorporate multiple SNPs. To make use of what was established for single-SNP logistic regression, the multiple-SNP models of Cordell and Clayton [2002] will now be presented, following by the multiple-SNP models of Balding [2006].

Multiple-SNP regression models, example 1

Proceeding from their single-SNP model that was presented earlier in this section, additional SNPs can be included in the regression model of Cordell and Clayton [2002]. Assume that there are three SNPs under study. The additive model for one SNP (3.5) can be duplicated for the three SNPs, resulting in the model

$$\ln\left(\frac{\pi}{1-\pi}\right) = \beta_0 + \sum_{i=1}^{6} \beta_i x_i \tag{3.25}$$

Here the pairs (x_1, x_2), (x_3, x_4), and (x_5, x_6) correspond to SNP 1, 2 and 3. The pairs follow the same coding as the variables x_2 and x_3 in (3.5) for each SNP, resulting in a model that accounts for the additive effect of each allele as well as dominance. The three $\beta_1 x_1$-terms from (3.5), which are constants, are replaced by a single β_0 term in (3.25). This model can then be compared to a model in which the third SNP is excluded from the model:

$$\ln\left(\frac{\pi}{1-\pi}\right) = \beta_0 + \sum_{i=1}^{4} \beta_i x_i \tag{3.26}$$

which is the same model under the constraint that $\beta_5 = \beta_6 = 0$. The models (3.25) and (3.26) are thus nested and can be compared using the same GLM procedure as was described for the single-SNP case earlier in this section, yielding a 2df test.

Multiple-SNP regression models, example 2

Let there be I SNPs. Introducing similar three indicator variables and βs for each SNP in the same fashion as in Equation (3.6), the model becomes

$$\begin{aligned} \ln\left(\frac{\pi}{1-\pi}\right) &= x^T\beta \\ &= I_{N_1N_1}\beta_{11} + I_{N_1M_1}\beta_{12} + I_{M_1M_1}\beta_{13} \\ &\quad + I_{N_2N_2}\beta_{21} + I_{N_2M_2}\beta_{22} + I_{M_2M_2}\beta_{23} \\ &\quad + \cdots + I_{N_IN_I}\beta_{I1} + I_{N_IM_I}\beta_{I2} + I_{M_IM_I}\beta_{I3} \\ &= \sum_{i=1}^{I} I_{N_iN_i}\beta_{i1} + I_{N_iN_i}\beta_{i2} + I_{N_iN_i}\beta_{i3} \end{aligned} \tag{3.27}$$

Like in the single-SNP case, restrictions can be put on the βs to test additive, recessive or dominant models which can then be compared to the full model, (3.27) in a GLM framework. For instance, for the additive model, Balding [2006] proposes requiring that every $H_0 : \beta_{i2} = \frac{1}{2}(\beta_{i1} + \beta_{i3})$, similar in spirit to what was done in the single-SNP case.

Number of	Number not rejected	Number rejected	
True null hypotheses	U	V	m_0
Non-true null hypotheses	T	S	m_1
total	$m - R$	R	m

Table 3.8: Notation of quantities associated with null hypotheses

3.6 Multiple testing

In population-based SNP/disease association, the problem of multiple testing occurs when one performs a large number of hypothesis tests for association between single SNPs and the disease. The problem is that while each individual hypothesis test controls its own type I error rate, which is the probability for a false positive SNP/disease association, a large number of hypothesis tests there will induce with a high probability one or more false positives. This is because each of the numerous individual tests has a small chance of resulting in a false positive. If post-association analyses are expensive, false positive SNPs can increase the costs of identifying the true positive disease-causing SNPs which was the object of the association study in the first place.

In multiple testing we focus on controlling various new versions of the type I error rate for multiple hypothesis tests [Dudoit et al., 2003]. These are related to the familiar type I error for a single hypothesis, which is the probability of a false positive, but apply to multiple tests instead of only one. The most popular type I error rate is the False Discovery Rate (FDR). Another version is the Family-wise Error Rate (FWER). There are various procedures for controlling a given error rate version. Dudoit et al. [2003] introduce these topics in the context of multi-array data resulting from the simultaneous measured expressions of a vast number of genes.

False Discovery Rate (FDR)

The False Discovery Rate (FDR) can be introduced in the following way [Dudoit et al., 2003]: There are m individual SNPs under study. For each SNP, we apply a hypothesis test (for instance, CA test for trend) for association of risk alleles of SNP with the disease. Notation for the situation can be found in Table 3.8 [Benjamini and Hochberg, 1995]. In Table 3.8, m and R are the total number of null hypotheses tested and the total number of observed rejections, respectively. Here m is a fixed number from the start of the experiment, while R is a random variable of which we observe a realisation. These are the only quantities in Table 3.8 that are known to us in the situation. The numbers m_0 and m_1 are the

number of true null hypotheses and false null hypotheses, respectively. These are unknown numbers that can also be viewed as fixed from the start of the experiment. The explanation of the last four quantities, S, T, U and V, is given in Table 3.8. Like R, these are random variables, but unlike R we do not observe any realisations of them. They remain unobserved and unknown to us in the experiment.

Table 3.8 can be related to the context of SNPs. If the hypothesis tests use the CA tests for trend statistic for single SNPs with H_0: $\beta = 0$ against H_1: $\beta \neq 0$ according to Subsection 3.2, then m_0 are those of the SNPs under study that actually have $\beta = 0$, which means they are unassociated with the disease. Likewise, m_1 are all those SNPs in the experiment that actually have $\beta \neq 0$ and are associated with the disease. These numbers are unknown to us. The only quantity that is known to us in addition to m is R, which is the observed number of CA tests that is actually rejected in this experiment, that is, SNPs that we suspect are associated with the disease.

Although not all of the quantities in Table 3.8 are observed or known, they can still be utilised by us in order to define relevant measures for the experiment. In particular, the quantity V appears interesting, since it is the unknown number of rejected true null hypotheses. It reminds us of the familiar type I error for a single hypothesis test, and it is also a part of the expression which is known as the False Discovery Rate (FDR):

$$FDR = E(Q),\ Q = \begin{cases} \frac{V}{R}, & R > 0 \\ 0, & R = 0 \end{cases}$$

The usage of the term 'type I error rate' can be motivated by the fact that V is the total number of type I errors, and $\frac{V}{R}$ is the proportion of this number to the total number of actual rejections. One advantage of the FDR compared to the other type I error rates mentioned in Dudoit et al. [2003] is that it enables a resulting list of candidate SNPs of which the expected proportion of false positives is know. This proportion is known, but not the identity of the false positives. As a simple example in the context of SNPs: Suppose a list of 105 candidate SNPs was produced, and by test design, a FDR of 0.05 was achieved. Then we would know that 5 false positive SNPs and 100 true positive SNPs would be expected. We would know nothing about the identity of the false positive SNPs, though.

Next, a procedure for controlling of the FDR will be considered.

Benjamini/Hochberg step-up procedure for strong controlling of FDR

Now assume that we have chosen the FDR as our measure of type I error rate. In the situation of m individual SNPs under study, resulting in m simultaneous hypothesis tests, we now want a procedure that can control the FDR, for instance keep it below a certain level α. The level α could be set to 0.05 or to another number. The Benjamini/Hochberg step-up procedure by Benjamini and Hochberg [1995] is one such procedure, for strong controlling of the FDR. The 'strong' refers to strong and weak control of type I rates in multiple testing, a concept which is discussed in detail by Dudoit et al. [2003]. I will now describe the procedure, based on the review by Dudoit et al. [2003]. The term 'step-up' procedure is used because the procedure starts with the smallest p-value, which is the p-value corresponding to the hypothesis that we would like the most to reject, and then includes successively larger p-values until a certain criterion is no longer fulfilled. We end up with a set of included p-values, and we reject the set of hypotheses that corresponds to that set of included p-values. By doing this, we achieve the desired value of FDR. In the words of Dudoit et al. [2003]:

> "Let $p_{r_1} \leq p_{r_2} \leq \cdots \leq p_{r_m}$ be the observed ordered (...) p-values. For control of the FDR at level α define $j^* = \max\{j : p_{r_j} \leq \left(\frac{j}{m}\right)\alpha\}$ and reject hypotheses H_{r_j} for $j = 1, ..., j^*$. If no such j^* exists, reject no hypothesis."

The required conditions and the proof for the Benjamini-Hochberg procedure are beyond the scope of this project.

Example of use of the algorithm

Assume that we desire for our experiment a FDR no greater than α. We observe all the p-values and order them. At first, $j = 1$, so we check if p_{r_1}, the smallest p-value, is less than $\frac{1}{m}\alpha$. If not, then we can reject no hypotheses at this FDR level. If it is, then we consider the next p-value and check if $p_{r_2} \leq \frac{2}{m}\alpha$. We see that the term on the right side of the inequality increases by $\frac{\alpha}{m}$ each time j increases by 1. If say p_{r_5} is the first p-value that fails to fulfill the inequality, then we reject the hypotheses corresponding to $\{p_{r_1}, p_{r_2}, p_{r_3}, p_{r_4}\}$ and no more.

3.7 Sample size and power

The statistical power of a test is associated with its probability to reject a null hypothesis, H_0, when the null hypothesis is false. In the context of SNP case-control association, the null hypothesis is that one or more SNPs are not associated with a disease. Therefore, power in this case is associated with the probability of detecting disease SNPs, depending on the null hypothesis used and given the assumptions from which the hypothesis test observator distribution was derived.

If only one SNP is examined and only one hypothesis test is made for that SNP, if the type II error for the hypothesis test, denoted by β is known, then the power of the test is $1-\beta$. If β is not known, then a possible way of assessing power in this case is the following: Assume that there has been obtained by a biologist a real case-control data set of SNP genotypes and disease status, with r cases and s controls, for instance on the form of Figure 3.2. Assume that previous studies have identified a disease SNP in the data set. Power for a specific hypothesis test for one SNP can be assessed empirically by simulating a number of data sets which mimic the real data set as closely as possible, then applying the hypothesis test once for each data set and recording whether the disease SNP was detected. If there are n_S data sets, then an empirical assessment of power is

$$\frac{\text{No. of times disease SNP was detected}}{n_S} \tag{3.28}$$

Other empirical assessments of power can include recording how often the p-value of the target SNP is smaller than a number, for instance 0.05. If there are also other SNPs that have significantly small p-values, one can examine how well the target SNP ranks among all those SNPs. Or, one may ask how often the p-value of the target SNP is among the lowest N p-values. If the parameters of the simulation and disease model are varied, such as the number of cases and controls, then these power considerations can provide indications about properties real data sets should have in order to increase power in the statistical analysis. This can provide a guide for assessing which expensive studies that should be considered not undertaken due to reduced probability of detecting SNPs.

Recalling the concerns about multiple testing from Section 3.6, if there are more than one SNP under examination for disease associations there will consequently be more than one hypothesis test performed. Then, using the notation of Table 3.8, a measure of a generalised power for multiple hypothesis tests is the expected value of non-true null hypotheses not rejected, divided by the total number of non-true null hypotheses: $\frac{E[S]}{m_1}$.

Chapter 4

Methods

4.1 Simulation using genomeSIM

The genomeSIM computer package is a tool that can simulate large-scale genomic data sets through forward-time population simulation [Dudek et al., 2006]. The user specifies the SNP allele frequencies of an initial population, which is subsequently crossed for a number of generations to produce a final population. Disease affection status is then assigned to the individuals of the final generation by checking their genotypes against the risk alleles of a prespecified disease model. A few lines of a genomeSIM data set is shown in Figure 3.2. Each row corresponds to an individual of the population. The first columns indicate disease affection status with the numbers 1 and 0. The other columns correspond to different SNP loci and indicate the number (2, 1 or 0) of copies of the disease allele in the genotype at each locus.

The genomeSIM forward-time simulation algorithm can be divided into three main steps. The steps and some of the input parameters are summarized in Table 4.1.

The genomeSIM algorithm

Step 1. Establishment of initial population

The initial population is established by choosing a population size and determining the SNP genotype at every SNP for all individuals in that population. The genome of each individual is represented by two chromosomes containing all the specified total number of genes (NUMGENS). Each gene is assigned a number of biallelic SNPs, randomly generated for each gene within a specified common minimum/-maximum gene count threshold parameter (MINSNP, MAXSNP). Each SNP is

Step	Important input parameters	
1. Establishment of initial population	POPSIZE:	Population size
	NUMGENS:	Number of genes
	MINSNP/MAXSNP:	Threshold for random numbers of SNPs on each gene
	ALLELELIMITS:	Minor allele frequencies for unspecified SNPs
	ALLELEFREQS:	Minor allele frequencies for specified SNPs
2. Random crossing of population for a number of generations	NUMGENS:	Number of generations
	MINRECOMB/ MAXRECOMB:	Threshold for random recombination frequencies for SNPs
3. Assignment of disease affection status to final population	MODELFILES:	Specifies disease model

Table 4.1: Main steps in genomeSIM algorithm

assigned an allele frequency number for its minor allele, also randomly generated within a specified common minimum/maximum threshold parameter, this time for allele frequencies (ALLELELIMITS). The user has the option to override the ALLELELIMITES parameter by specifying individual allele frequencies for chosen SNPs using the ALLELEFREQS parameter. Similarly, each SNP is assigned a recombination fraction number for recombination between itself and the next adjacent SNP in that gene, randomly generated according to the common minimum/maximum recombination frequency threshold parameter (MINRECOMB, MAXRECOMB). To quote Dudek et al. [2006],

> Thus, all recombination fractions are random and independent. SNPs are unlinked across genes.

The concept in genomeSIM of SNPs being unlinked across genes is a simplified recombination approach because in real life, loci can be linked, for instance, but not always, due to correlation of recombination frequencies with physical distance between the loci [Hartl and Jones, 2006, pg. 132]. Recombination is not relevant for the initial population, but becomes relevant in Step 2. Once the allele frequencies are determined, the genotype is randomly generated for each SNP. Adding genotypes from the two chromosomes, this results in either 0, 1 or 2 copies of minor alleles for every SNP.

Step 2. Random crossing for a number of generations

Once the initial population is established, the simulator proceeds to advance that population the prespecified number of generations ahead in time. A new generation is formed by producing a total number of new individuals equal to the size of the parental population. After a new generation is formed, the old generation is forgotten and never considered again. Hence, the population size is constant throughout time. When a new individual is to be formed, two parents are drawn with replacement from the parental population and produce one gamete each which are then joined to form the new individual. In genomeSIM, a parent produces a haploid gamete by crossing over his two chromosomes and passing one of the resulting altered chromosomes to the gamete. In Step 1, each SNP was assigned a random recombination fraction number using the minimum/maximum threshold parameters (MINRECOMB, MAXRECOMB). Quoting Dudek et al. [2006],

> A crossover is conducted as follows. genomeSIM selects one chromosome to be the start chromosome and begins copying allele values from that chromosome into the new chromosome. At every interval between SNPs, the simulator checks the recombination fraction against a randomly generated number. When the number is less than or equal to the

fraction, the simulator switches chromosomes (assuming independent assortment) and begins taking allele values from the second chromosome. The simulator continues to check each interval and copies the allele values for the current chromosome until it reaches the end of the genome or another crossover takes place.

Step 3. Assignment of disease affection status to final population

When the final generation is reached, the simulator assigns disease affection status to the individuals in that generation by comparing their risk SNP genotypes with a penetrance table. If there is only one risk SNP, then the penetrance table is the vector $[f_0, f_1, f_2]$ of penetrances from Table 3.3. If there are two risk SNPs, then the penetrance table becomes a matrix $[f_{ij}]$, $i, j = 0, 1, 2$. The penetrance table must be provided at the start of the simulation in the MODELFILE parameter. One has the option of not providing a penetrance table and instead have the simulator assign disease affection completely at random.

4.2 Framework in R for pre-study sample size and power considerations using genomeSIM simulation

Simulation programs such as the genomeSIM package enable the construction of simulating artificial genomic SNP data sets. We may use such simulated data as the starting point for performing statistical analyses and assessing power and sample size issues. It is hoped that insights will result from this analysis of artificial data sets, which in turn can be applied to the analysis of real data. For instance, a biologist may request assessments on whether a given number of cases and controls are enough for detecting a suspected disease SNP with a given method. By varying the sample size in the simulated data, something which is not as easily achieved with real data, statistical power can be estimated as a function of sample size. The validity of any application of framework results to real data depends on how well the simulated data mimic real data and on the assumptions of the analysis. In this section a framework is presented which combines genomeSIM and R functions into a single flexible tool that can be used for genomic simulated case-control analyses.

An overview of the framework and its components is presented in Figure 4.1. Broadly, the framework consists of five steps:

1. Given a requested number of cases and controls, estimate the required population size for obtaining the desired number (of cases and controls) for a

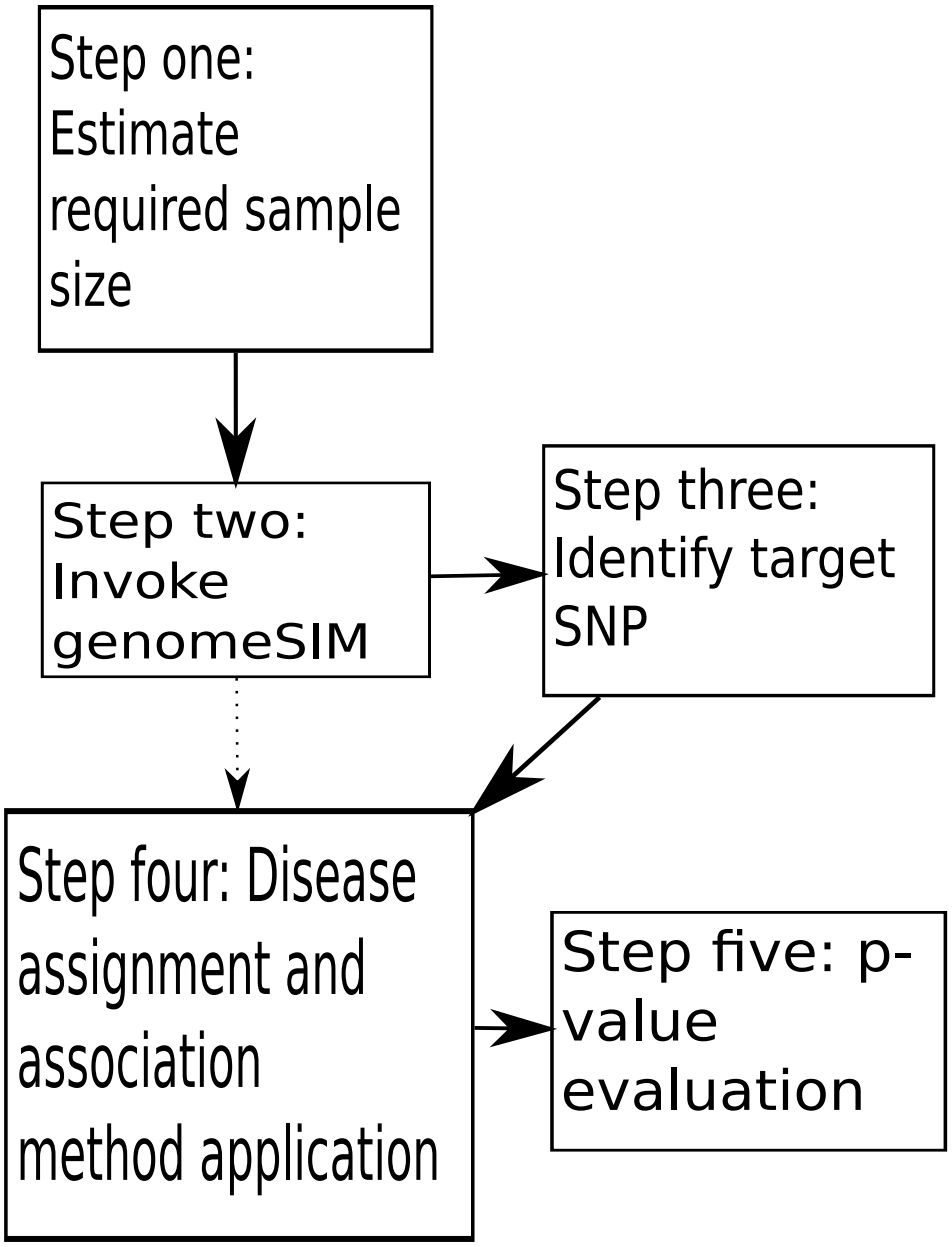

Figure 4.1: Diagram showing assumed flow of work within framework

given disease model.

2. Invoke genomeSIM, using the population size that was estimated in the previous step. Simulate population data set of genomes without disease assignment. (The native genomeSIM disease assignment is removed from the data set.)

3. Identify one (or several) SNP in the population genomes which matches a desired allele frequency

4. On the basis of the one dataset SNP genotypes and the identified SNP, assign disease status a number of times, producing multiple case-control datasets. Apply one or more SNP/disease association methods on each of these datasets, aiming on detecting the specified SNP.

5. Apply one or more performance measures on the outcome. This could include comparing the resulting p-values of the specified SNP with the p-values of the other SNPs.

The framework is structured as a set of R functions which can be used independently of each other. In 'script mode', a script is delivered with the framework that performs the five steps automatically based on default values which can be edited in the script. However, because of the modular structure with separate R functions, one has an alternative option of 'flexible mode' where one can try out each of the R functions in the framework manually, trying out different values in each step before advancing to the next. This allows for greater flexibility.

Step 1: Estimating required population size

The number of cases and controls in the final generation of the simulated data set will be a random variable which depends on the penetrance table and the population size used in the simulation. In a genomeSIM simulation, given a penetrance table and the allele frequencies of one or more risk SNPs that one wants to simulate, it may be desirable to estimate a lower limit n of the population size required for obtaining at least r cases and/or at least s controls. In a one-SNP example, this may be done as follows: From Table 3.3, the prevalence of disease is $K = \sum_{i=0}^{2} g_i f_i$ so, assuming n independent Bernoulli trials, the expected number of cases is $r = nK = n\sum_{i=0}^{2} g_i f_i$. The expected number of controls is $s = n(1-K) = n(1-\sum_{i=0}^{2} g_i f_i)$. Hence, the lower limit of the population size required is

$$n = \begin{cases} \frac{r}{\sum_{i=0}^{2} g_i f_i} & \text{for at least } r \text{ cases} \\ \frac{s}{1-\sum_{i=0}^{2} g_i f_i} & \text{for at least } s \text{ controls} \\ \max\{\frac{r}{\sum_{i=0}^{2} g_i f_i}, \frac{s}{1-\sum_{i=0}^{2} g_i f_i}\} & \text{The maximum, for at least } r \text{ cases and } s \text{ controls} \end{cases} \tag{4.1}$$

If the Hardy-Weinberg equilibrum holds, then the SNP genotype frequencies g_i in (4.1) can be replaced by the SNP allele frequency expressions from Equation (3.1).

Step 2: Invoke genomeSIM

From R, genomeSIM is invoked. For convenience, the configuration file that must be used with genomeSIM is automatically edited by the framework, based on the values that are passed by the user into the R function. In the current version of the R function, only the population size parameter is edited in this way. All the remaining simulation parameters in the configuration file have default values. However, the framework allows for the user to provide a pre-edited genomeSIM configuration file and use that one instead.

In the current version, every SNP in the framework is simulated based on predetermined random initial allele frequencies ranging between $g = 0.05$ and 0.5. (Recall from Section 4.1 the ALLELELIMITES parameter.) The outcomes of the initial allele frequency assignments are stored in a file. However, as the simulated population is randomly crossed ahead in time, the SNP allele frequencies may change, so it is not guaranteed that the allele frequencies of the ultimate data set matches those that were set as initial frequencies. This becomes relevant in the next step.

Step 3: Identify target SNP

With the completion of Step 2, a data set of SNP genotypes is obtained. The next step is to identify a SNP as the disease SNP on which later disease status assignment will be based. In order to obtain the expected number of cases and controls that was calculated in Step 1, one has to make sure that the target SNP has the right allele frequency, g. The target SNP is identified in the following way. Since one does not know the final g for the simulated SNPs, one must instead estimate g_0, g_1 and g_2 based on allele counts. For a given SNP,

$$\hat{g}_i = \frac{\text{No. of individuals with } i \text{ copies of } M \text{ allele in genotype of that SNP}}{\text{Total no. of individuals}} \tag{4.2}$$

Then, since the genomeSIM simulation is performed assuming Hardy-Weinberg equilibrium, three different estimates of g can be obtained based on g_0, g_1 and g_2 from (3.1). The three estimates, denoted by $\hat{g}^{(0)}$, $\hat{g}^{(1)}$, and $\hat{g}^{(2)}$, are:

$$\begin{aligned} g_0 = (1-g)^2 &\Rightarrow \hat{g}^{(0)} = 1 - \sqrt{\hat{g}_0} \\ g_1 = 2g(1-g) &\Rightarrow \hat{g}^{(1)} = \frac{1}{2} - \frac{1}{2}\sqrt{1-2\hat{g}_1} \\ g_2 = g^2 &\Rightarrow \hat{g}^{(2)} = \sqrt{\hat{g}_2} \end{aligned} \tag{4.3}$$

In (4.3), the solutions $\hat{g}^{(0)} = 1 + \sqrt{\hat{g}_0}$ and $\hat{g}^{(2)} = -\sqrt{\hat{g}_2}$ are rejected because $\hat{g}$ must be a probability, $0 \leq \hat{g}^{(1)}, \hat{g}^{(2)} \leq 1$. The solution $\hat{g}^{(1)} = \frac{1}{2} + \frac{1}{2}\sqrt{1-2\hat{g}_1}$ is rejected because it was assumed in the simulation that g was the allele frequency of the least common allele, so that $0 \leq g \leq \frac{1}{2}$.

Based on the three estimates[1] in (4.3) for each SNP, the framework must decide which SNP that has the allele frequency closest to the desired g. In the framework, this is currently done by minimising the difference between g and the mean of the estimates in absolute values. If there are m SNPs, and $g^{(ij)}$ is the ith estimate of g for the jth SNP, then the chosen estimate $\hat{g}$ is

$$\hat{g} = \text{argmin}_j \left[g - \frac{1}{3}\sum_{i=1}^{3} |\hat{g}^{(ij)}| \right], i = 0, 1, 2,\ j = 1, ..., m \tag{4.4}$$

That however is only one of many possible choices for such a decision method. Perhaps the estimates could be weighted if some of the estimates were assumed to be more accurate than the others.

Step 4: Disease assignment and association method application

In this step, given a simulated genomeSIM data set and a target SNP, disease status is assigned to all the individuals in the data set multiple times based on their genotype for the target SNP. The penetrances f_0, f_1 and f_2 (see Section 3.1) are used for determining disease status for an individual based on the genotype. If the genotype of the individual is NM, for example, then a random number from the uniform[0,1] distribution is created and disease assigned if the number is less than

[1]The estimate $\hat{g}^{(1)}$ in (4.3) becomes imaginary if $\hat{g}_1 > \frac{1}{2}$. In (4.3), the derivative $\frac{dg_1}{dg}$ is negative for $g > \frac{1}{2}$ and positive for $g < \frac{1}{2}$. The maximum of g_1 is reached in $g = \frac{1}{2}$ with $g_1 = \frac{1}{2}$ as the maximum value. But although g_1 never exceeds $\frac{1}{2}$, its estimate $\hat{g}_1$ from (4.2) could, which would produce imaginary values for $\hat{g}^{(1)}$.

f_2. As a result, several different case-control data sets are obtained based on the single initial population genomes data set. These case-control data sets can then be analysed in R with SNP/disease association methods. If disease is assigned n_S times, the result will be n_S sets of p-values for all the SNPs. Currently, the framework uses the *maxstat* method from the SNPassoc R package (See Gonzalez et al. [2008] for details). This choice of method can be changed at a later time according to the needs of the users of the framework.

In the current version of the framework, although the numbers of cases and controls are specified by the user, the resulting ratio of cases of controls is not controlled, because the framework simulates cases and controls in order to provide *at least* that many cases *and at least* that many controls. Thus, there could be too many of either cases or controls.

Step 5: Performance measurement

In the final step, the p-values and other performance measurements of interest resulting from the previous step will be evaluated.

4.3 Suggested application example and conceptual test

Here an application of the framework is presented. Assume the following:

- One disease SNP under study has an allele frequency g and there are a total of m SNPs.
- There is a dominant disease model and the odds ratio for the dominant case, Equation (3.22), has a known value, for instance $\psi = 2$.
- There is available a method for detecting SNP/disease associations under a dominant model. (e.g. the *maxstat* method.)
- The prevalence of the disease is K.

Assuming direct association, how many cases and controls are needed on average to achieve with probability $\beta = 0.9$ that the target SNP will appear among the N lowest p-values that have significant low values?

This is an example of a first-step empirical analysis on a path towards a more realistic model, using the framework that has been presented in this thesis.

Example Step 1: Estimating required population size

We know g and need to find penetrances f_0, f_1 and f_2 so that we can estimate the required population size for obtaining the desired number of cases and controls. We utilise the odds ratio given in Equation (3.22) to find the penetrances. If we assume that for the dominant model $f_1 = f_2 \equiv f_D$, then Equation (3.22) reduces to

$$\psi = \frac{\frac{f_D}{1-f_D}}{\frac{f_0}{1-f_0}} \tag{4.5}$$

This is an equation with two unknowns, f_D and f_0, so we need more information in order to arrive at a unique solution. We assume that the prevalence $K = \sum_{i=0}^{2} g_i f_i$ of disease is known and provides the equation needed. Applying the Hardy-Weinberg expressions for g_i (3.1) and $f_1 = f_2 \equiv f_D$, the prevalence becomes

$$K = (1-g)^2 f_0 + [2g(1-g) + g^2] f_D = (1-g)^2 f_0 + (2g - g^2) f_D \tag{4.6}$$

Equations (4.5) and (4.6) are solved to yield the penetrances. The solution is

$$\begin{aligned} f_D &= \frac{\sqrt{b^2 - 4ac}}{2a} \\ f_0 &= \frac{K - (2g - g^2) f_D}{1 - g^2} \end{aligned} \tag{4.7}$$

with $a = (2g - g^2)(\psi - 1)$, $b = [\psi(g^2 - 2g - K) - (1 - g^2 - K)]$ and $c = \psi K$.

With these solved values for f_D and f_0, the required lower limit to the simulation population size for obtaining any number of cases and controls can be computed using Equation (4.1), which is implemented in one of the R functions in the framework.

Example Step 2: Invoke genomeSIM

Here genomeSIM is evoked, producing the data set of SNP genotypes which is expected to yield the desired number of cases and controls in Step 4.

Example Step 3: Detect target SNP

One of the R functions in the framework finds a SNP in the population genomes data set which will serve as the disease SNP in the next step. (See Equation 4.4.)

Example Step 4: Disease assignment and association method application

Using one of the R functions in the framework, disease status is assigned to the population based on their genotypes of the disease SNP. This is repeated n times producing n_S different case-control dataset from the initial population genomes data set, and it is expected that at least the desired number of cases and controls is produced every time. In the same loop the dominant SNP/disease association method is also applied, yielding n_S sets of p-values, each set consisting of m p-values since there are m SNPs.

Example Step 5: Performance measurement

The frequency β with which the disease SNP is found among the N lowest p-values that are significally low is computed by one of the R functions in the framework. If $\beta < 0.9$, then the number of cases and controls will have to be increased, thus requiring a new simulation starting with Step 1 to estimate the new required simulation population size.

The number of cases and controls can be increased incrementally using a looped script until β reaches its desired value (0.9 in this case). The number of cases and controls which was used in the last loop before β reached 0.9 can be taken as an empirical indication of the number of cases and controls required for a similar study using real data.

Test of framework

A conceptual test is made in order to determine whether the suggested application could be achievable with the framework. Assuming a dominant disease model with a set of dominant penetrances, and using the *maxstat* R function, the goal is to determine whether successive simulations by the framework with increased sample sizes (i.e. number of cases and controls) will deliver increased power of detecting the disease SNP. Unlike in the application, the test does not use a specified odds ratio to calculate the penetrances. Instead, penetrance values are chosen specifically for the test. The test performs five instances of the scheme in Figure 4.1 with five different sample sizes ranging from (100 cases, 200 controls) to (2000 cases, 4000 controls). The parameters that were common for all five instances are shown in Table 4.2.

Parameters that varied from instance to instance are shown in Table 4.3.

The choice of the common penetrance value of $(f_0, f_1, f_2) = (0.3, 0.35, 0.35)$ reflects a dominant disease model since the disease risk of genotypes NM and MM

Parameter	Value
f_0, f_1, f_2	0.3, 0.35, 0.35
g	0.4
No. of genes	20
SNPs per gene	5
Total no. of SNPs	100

Table 4.2: Parameters that were common for all five instances

Instance	Parameter values
1	100 cases, 200 controls, $n_S = 100$, set.seed(121)
2	300 cases, 600 controls, $n_S = 50$, set.seed(122)
3	500 cases, 1000 controls, $n_S = 100$, set.seed(123)
4	1000 cases, 2000 controls, $n_S = 100$, set.seed(121)
5	2000 cases, 4000 controls, $n_S = 100$, set.seed(519)

Table 4.3: Parameters that varied for all five instances

are similar and higher than the disease risk of genotype NN. The choice was also motivated by the desire for a 1:2 ratio between cases and controls. (See discussion of this ratio in Step 4 in Section 4.2.) The choice of penetrances in the test provides approximately a 1:2 ratio between cases and controls.

The seeds set by R allow for reproduction of the simulation results since the computer's pseudorandom sequence starting point is identified. The seeds are different for some of the five instances due to an unstability in the chosen association method, the *maxstat* R function, when used with this framework. The reported seeds are seeds that were found to work with the five instances. There were other seeds that caused the *maxstat* method to crash. Also, the probability of crash increases with n_S because the *maxstat* R function is called $n_S \times 100$ times for each instance. In particular, the sample size used instance 2 was unstable. A number of seeds were tried in instance 2 with $n_S = 100$ disease assignments, without success. The seed reported in Table 4.3 worked when n_S was reduced to 50, hence the reduced n_S. We will report the instability to the maintainers of the SNPassoc package so that they can determine if it is a bug. In the meantime, this temporary problem can be remedied in the framework by replacing the *maxstat* function with another or by reducing n_S.

Chapter 5

Results

Figures 5.1, 5.2, 5.3, 5.4, and 5.5 show the results from the five simulated scenarios with parameters described in Tables 4.2 and 4.3.

For each scenario there are three pairs of plots. The first pair corresponds to the disease SNP that was identified in the framework's Step 3, the two other pairs correspond to random SNPs from the simulated data set that are used as reference SNPs. The left plot of every pair are histograms that show the frequencies of p-values in ranges from 0 to 1 with intervals of 0.05. In scenario 1, with the lowest sample size, 12 out of 100 p-values fall within the range of 0-0.05. The number of p-values in this range increases steadily as the the number of cases and controls are increased in scenarios 2-5, with scenario 5 yielding more than 80 out of 100 p-values in this range.

Likewise, the plot to the right of every pair displays the rate at which the SNP has a p-value which is among the N lowest p-values for that scenario. The upper right plot in Figure 5.1, for example, indicates that with 100 cases and 200 controls, the disease SNP has a rate of occurrence among the 40 lowest p-values of about 0.6.

For $N = 40$, the rate was calculated as follows: In each of the $n = 100$ disease assignments and applications of the *maxstat* method, every SNP in the simulated data set produced a p-value. An indicator variable I_i took the value 1 if the disease SNP was among the 40 lowest p-values and 0 otherwise, $i = 1, ..., 100$. The rate of occurrence is the mean of the $n = 100$ indicator values, $\frac{1}{100} \sum_{i=1}^{100} I_i$. The rate is calculated similarly for every other N, resulting in the plotted values.

In Figures 5.1 to 5.5 we see that the random reference SNPs have rates that are approximately linear. This suggests that they have uniform[0,1] p-value distributions. The rate of the disease SNPs deviate increasingly from the straight line

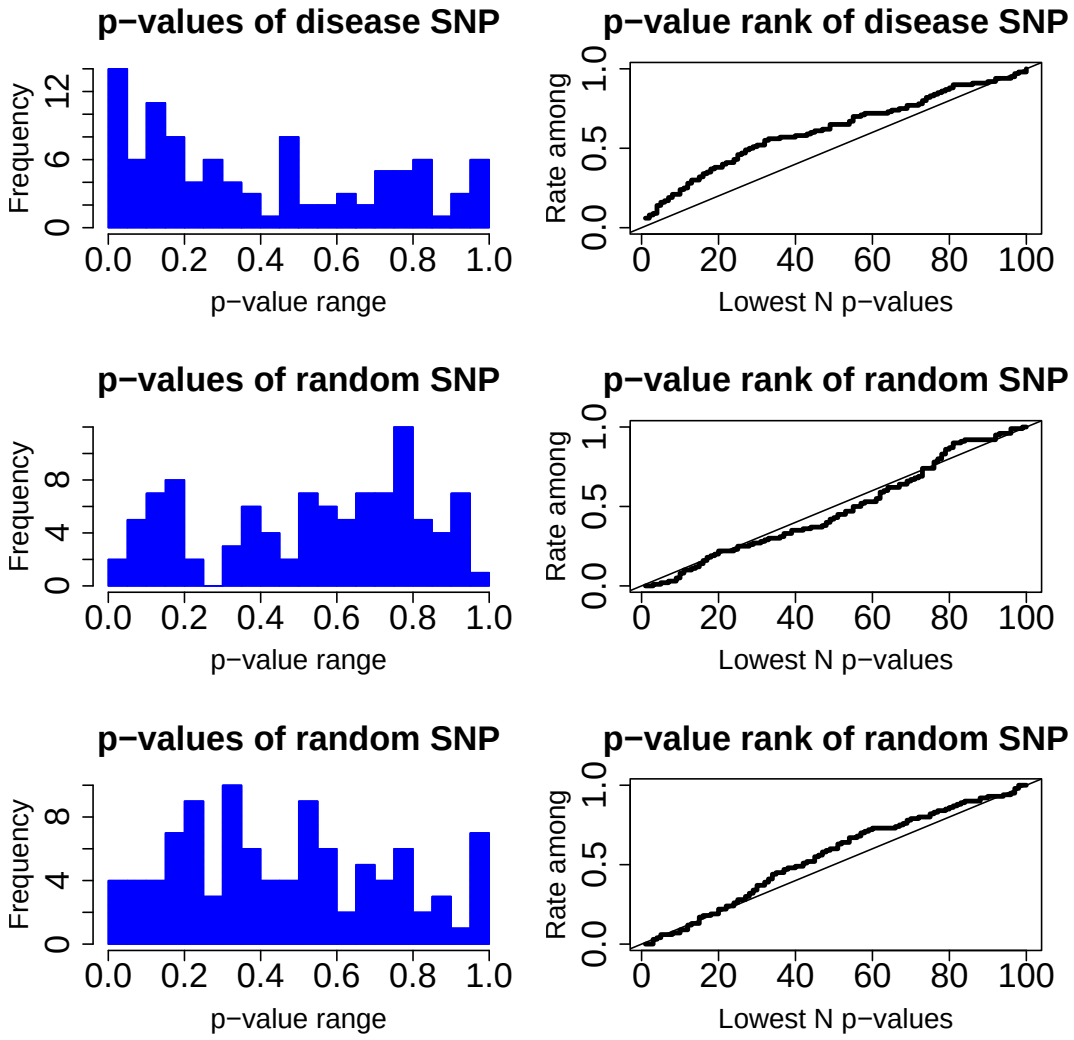

Figure 5.1: Results, scenario 1 (On avg. 99.57 cases, 202.43 controls)

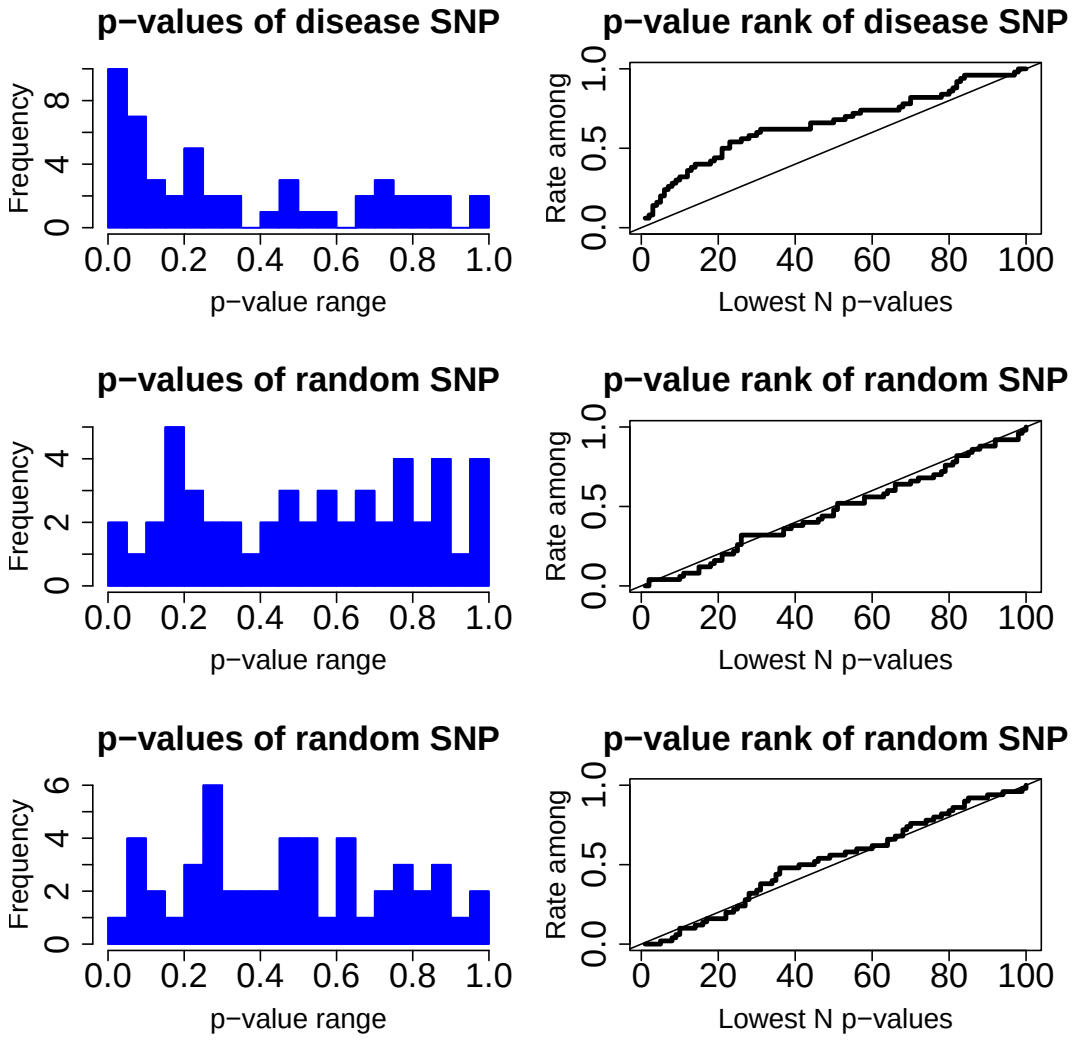

Figure 5.2: Results, scenario 2 (On avg. 303.08 cases, 600.92 controls)

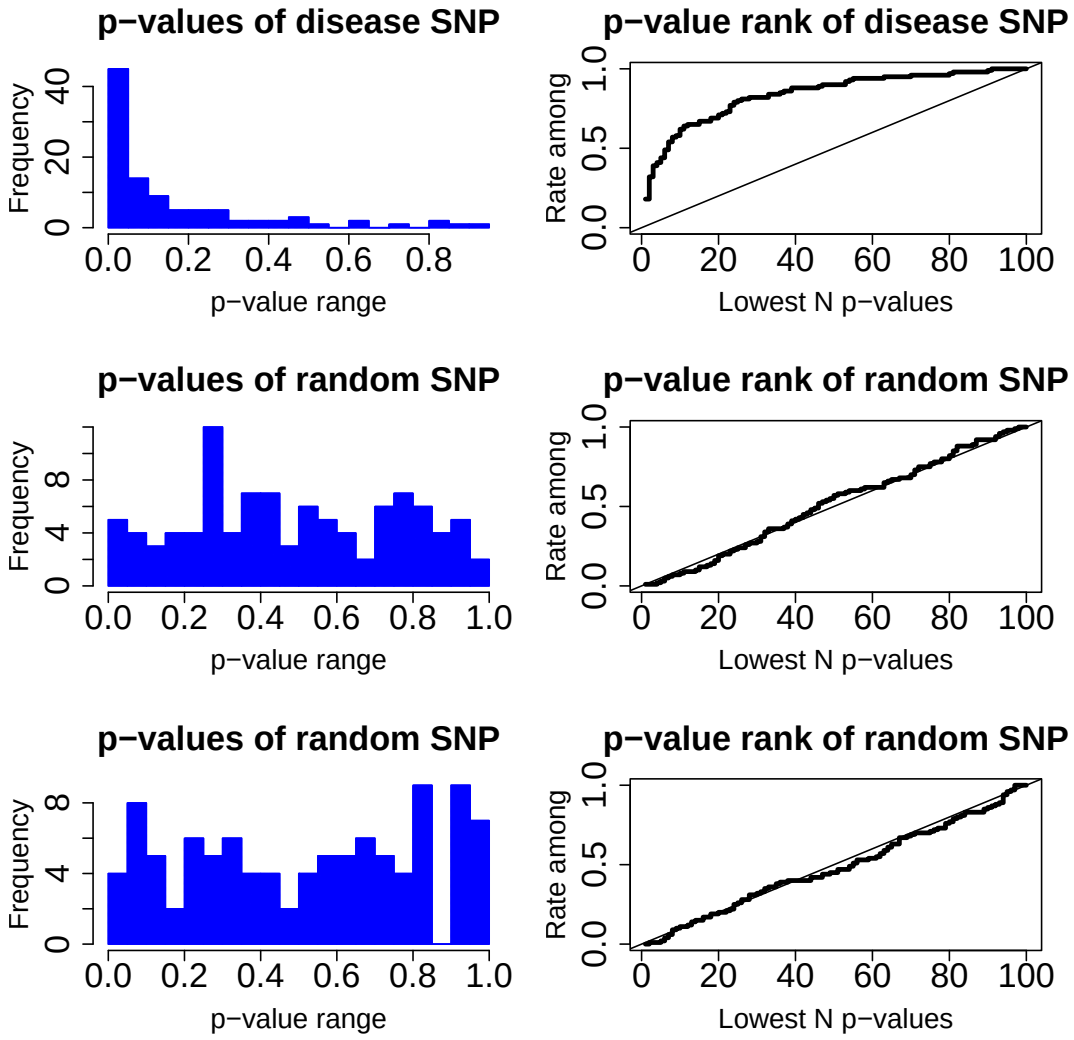

Figure 5.3: Results, scenario 3 (On avg. 498.82 cases, 1008.18 controls)

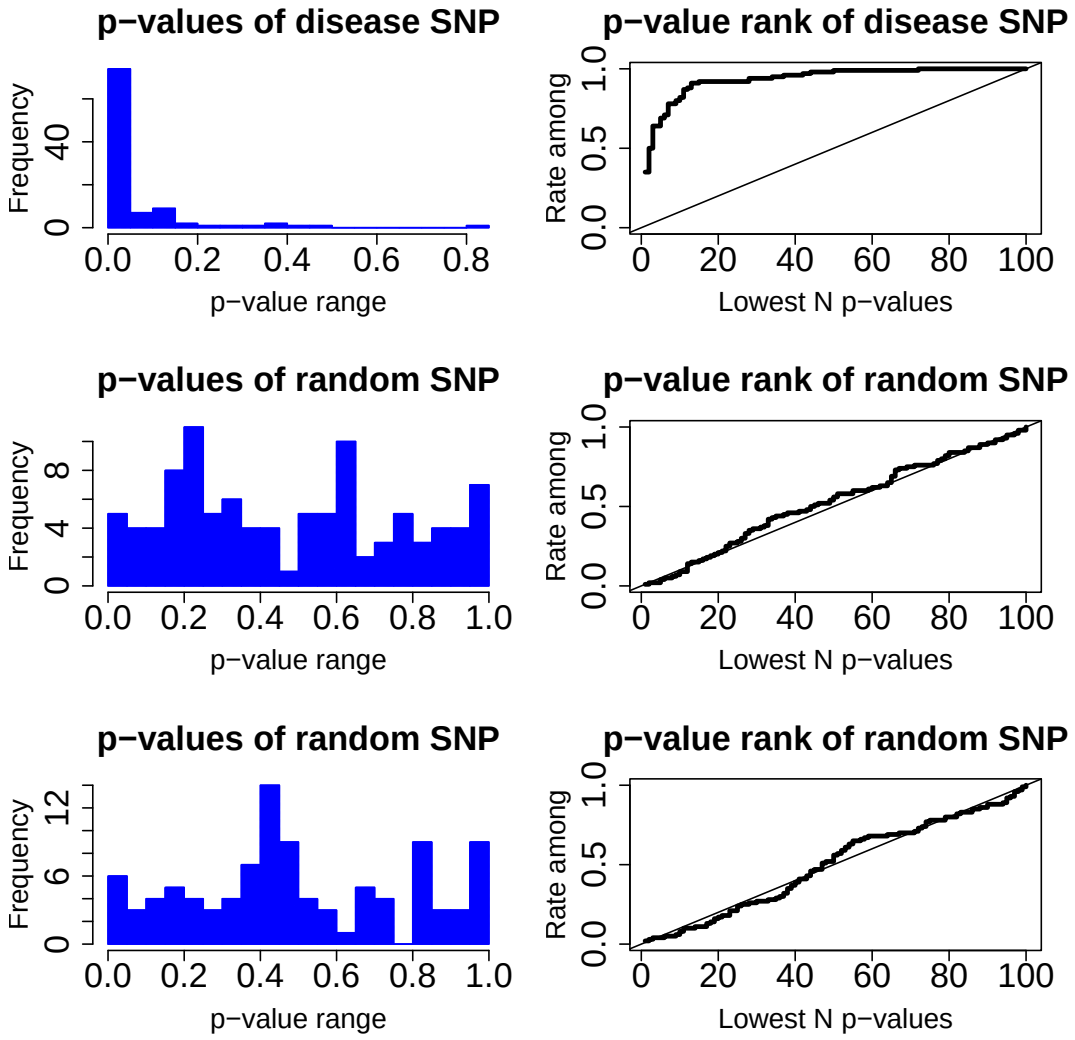

Figure 5.4: Results, scenario 4 (On avg. 997.34 cases, 2015.66 controls)

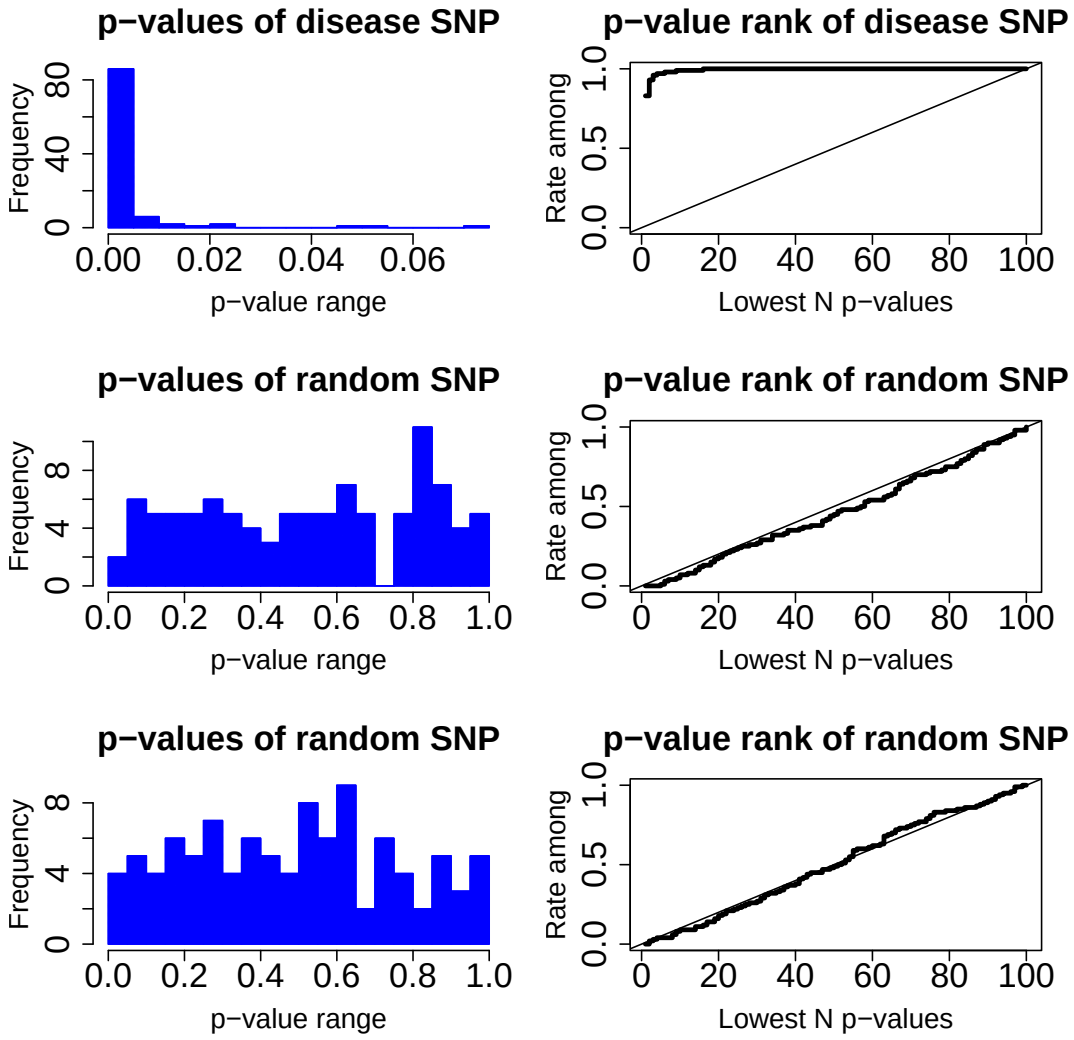

Figure 5.5: Results, scenario 5 (On avg. 2002.37 cases, 4022.63 controls)

as the sample size increases. As sample size increases, the rate reaches high values sooner, which indicates that the p-values of the disease SNPs are increasingly found among the lowest p-values. This is also indicated by the histograms.

In the spirit of the suggested application example in Section 4.3, if a hypothetical request had been made for an assessment of the sample size needed to ensure with probability $\beta = 0.9$ that the p-value of the disease SNP was found among the lowest y=10 p-values, then Figures 5.1-5.5 can be used to assess the needed sample size. It appears from Figure 5.4 that by finding $N = 10$ on the x axis, the rate is between 0.7 and 0.8, which indicates that 1000 cases and 2000 controls are not sufficient. In Figure 5.5, the rate is well above 0.9, which indicates that 2000 cases and 4000 controls may be too much. The sufficient sample size appears to be somewhere between 1000-2000 cases and 2000-4000 controls.

The simulation time for one scenario simulation was about 10 minutes on a computer with an Intel(R) Xeon(TM) CPU 3.20 GHz Mhz processor and 2 GB RAM.

Chapter 6

Discussion

The conceptual test of Section 4.3 revealed that in its current state, our simulation-based framework can be used according to its intention of making simulation-based sample size and power considerations for SNP case-control association studies. The immediate usage potential of the framework and its limitations will be discussed, as well as proposals for results that can be obtained with the framework if further adjustments are made.

On applying the simulation-based framework in its current state

By acquiring the R-code of its current state, anyone can make use of the framework to produce immediate results, given that their computer supports R and the genomeSIM program. It may be useful to biologists, who would provide the parameters relevant for their studies (e.g. assumed penetrances f_0, f_1, f_2, and an allele frequency g) and retrieve a recommended number of cases and controls, or the probability of their one disease SNP to be among a top specified number of detected SNPs. This may be done without modifications to the code. However, biologists should bear in mind the limitations of the framework in its current unmodified form and judge for themselves if the limitations are too severe for their application. Currently, only a single biallelic SNP is supported. In addition, results provided by the framework could be erranous if the simulated genomic data does not mimic the real genomic data well. In the framework, due to its modular structure the means of obtaining the simulated genomic data (genomeSIM) can be replaced easily by another simulation program, if a better one is available. The genomeSIM package also allows for more realistic simulations if the biologists specify more parameters beyond the defaults specified in the framework. The cost and effort of obtaining a better simulation program, such as the successor of genomeSIM, genomeSIMLA, could be weighted against the benefit on results of more realistic simulations. Even if a more realistic simulation program was ob-

tained, the results are not guaranteed to improve. In any case, the current state of the framework represents the first step towards more realistic analyses.

Another component of the framework that is easily replaced with little labor due to its modular structure is the chosen R statistical association analysis method, *maxstat*. Any R function that accepts data on the form of Figure 3.2 and is able to produce p-values for each SNP will do. There could be existing functions that have been implemented by others, if not, the user could implement the function. The flexibility of method replacement is useful for biologists whose projects involve specific favored methods.

On a computer with standard specifications, framework time consumption was moderate (10 minutes) for a scenario with 100 SNPs and a population ranging from 300 to 6000 individuals and a combined step of disease assignment and association method execution being applied n_S=100 times. The genomeSIM simulation program is written in ANSI C++ [Dudek et al., 2006], and the rest of the framework consists of R-code. In the scenario examples in this thesis, the genomeSIM simulations constituted only a few seconds of the total framework time consumption. In genome-wide studies, the number of SNPs can easily range in the ten-thousands, even hundred-thousands. This would increase framework time consumption due to both genomeSIM simulation, whose code cannot easily be improved, and also the time consumption of the R-code in the framework significantly unless the current code is improved. Also, if a more realistic data set was to be obtained, the number of mating generations in genomeSIM could be increased, increasing simulation time.

The sources for time consumption could be divided into three: Simulation program (e.g. genomeSIM), framework R functions for disease assignment, data loading and general setting up, such as the ones in the appendix of this thesis, and invoked R association analysis functions (e.g. *maxstat*). Currently, for each scenario, one and only one data set of SNP genotypes is simulated. Based on this one data set, disease status is repeatedly assigned, producing repeated case-control data sets. For each of these case-control data sets, the association method is applied once. Based on the duration of each of the three main sources of time consumption, the framework could be modified in order to reduce total time consumption. For instance, if the simulation program consumes very little time, like it did in the conceptual test in Section 4.3, then instead of invoking the simulation program only once a new data set of genotypes could be simulated for each of the repeated case-control data sets. In that case, the variation between each of the simulated genotype data set becomes relevant. If the variation is very high, for

instance due to a wide range of allele frequencies in genomeSIM due to a wide ALLELEFREQS parameter, then maybe the total number of such simulated case-control data sets should be increase in order to level out the variation.

If the main source of time consumption is the framework utility R functions, care could be taken in the general code design to increase the efficiency of the code, or a faster programming language such as C could be used. In fact, the framework R function for disease application is supported by genomeSIM, meaning that disease assignment can be done within its fast code. However, implementing the disease assignment in R allows for greater control and flexibility, rather than relying on a pre-bundled package such as genomeSIM. If both control/flexibility in disease assignment and efficient code is desirable, then flexible disease assignment could be implemented by the user in an efficient language and included into the framework.

If both genotype data simulation, framework R-code and invoking of association analysis R function are efficient and fast, but genotype data simulated is a bit slower than the other two, then another option could be to run a loop of genotype data simulations, and for each iteration in the loop a subloop could be ran with a modest number of disease assignments and association method applications.

The ratio of cases versus controls

The ratio of cases versus controls produced by the framework was mentioned in Section 4.2. In its current design, the framework uses genomeSIM to simulate the SNP genotypes of a whole population of individuals. Then, disease status is assigned based on specified penetrances f_0, f_1, and f_2 of a single SNP, producing cases and controls. In the current configuration, the framework does not control the ratio of cases versus controls. The resulting numbers of cases and controls reflect the prevalence K of disease in the simulated population using the disease model specified. Naturally, the resulting numbers of cases could differ from the ones desired by the user. For instance, if the penetrances are very low, then the prevalence K in Table 3.1 will be also be small, resulting in ratio of cases to controls which is too low. The framework can be modified in a later version to control the ratio by removing redundant cases or controls from the data set every time after disease status has been determined for all the individuals.

In the framework conceptual test in section 4.3, values $(f_0, f_1, f_2) = (0.3, 0.35, 0.35)$ were chosen in order to ensure a 1:2 ratio of cases versus controls because the lack of redundant case/control removal in the current version. If it is desirable to use the framework with a single SNP with a more modest effect and hence lower pen-

etrances, the penetrance values that were used in the example could be too high and the example unrealistic. Also, an allele frequency of $g = 0.4$ was used in order to obtain more than just a few number of individuals with the MM genotype since the frequency of the MM genotype is g^2 when Hardy-Weinberg equilibrium is assumed. This demonstrates that if a lower g and lower penetrancet are desired, case/control ratio controlling by removal of redundant cases or controls could be convenient.

Proposal for further applications and studies using the framework

Previously, the potential for immediate results by using the framework in its current state, possibly with only minor changes, was discussed. However, by modifying the framework beyond a mere replacement of modules, further possibilities for results can be possible. The step of going from one SNP to multiple SNPs is a more advanced modification which requires biological understanding of the how multiple SNPs can act together in causing complex diseases. In Sections 3.4 and 3.5, some multiple-SNP genotype concepts was explored as well as a conceptual study of how logistic regression can be applied to multiple-SNP case-control association. In order to incorporate multiple-SNP models into the framework, one needs to specify penetrance tables or another way of determining disease status of an individual based on his multiple disease SNP genotypes. Then one needs to replace the single-SNP association method with a multiple-SNP association method. Since many diseases are believed to be complex and affected by many SNPs, multiple-SNP expansion of the framework could be very useful. The same questions can be raised about whether the framework can realistically simulate scenarios of multiple SNPs realistically as were raised for the single-SNP scenario, especially since multiple-SNP models are more complex and thus more difficult to simulated realistically. Again, the framework with its exploratory potentials is a first step towards more realistic tools.

Similarly, indirect association methods could be incorporated into the framework, enabling the detection of other SNPs that are correlated with a causal SNP due to linkage disequilibrium.

Another modification which is more advanced than mere module replacement is incorporation of multiple testing. The plots of the random reference SNPs in the conceptual test of the framework in Section 4.3 indicated that the p-values of the reference SNPs had approximately uniform[0,1] distributions. Hence, as was also described in Section 3.6, false positive associations could arise due to chance. By incorporating for instance adjusted p-values due to false positives and controlling

type I error rates such as FDR and FWER, the framework could take into account the issue of multiple testing. In this thesis, multiple testing features have not been in the scope during construction of the framework. Focusing on the MAX test method, a related master study [Risberg, 2008] that has peered deeper into multiple testing aspects regarding this method suggests that the correlation between SNPs probably requires resampling. The study indicates that resampling of the FWER is too conservative and that resampling of the FDR might be better.

In the current version of the framework, p-values are obtained by designating one specific SNP as the disease SNP, then using its penetrance values to assign disease to all individuals and an association method to produce p-values for all SNPs. If multiple testing is to be incorporated in a later version of the framework, the change could occur at the level of the produced p-values by introducing adjusted p-values at that level. This does not constitute a very broad transition. Similarly, in the performance evaluation module of the framework, measures of average power can be introduced in place of regular power considerations at this level. These features could be added to the framework in order to advance it on the path from an initial approach towards more a more realistic tool.

Further results can be obtained if one considers the prospect of other sources of data and projects to be used in combination with the framework. For instance, HUNT (Helseundersokelsen i Nord-Trondelag) represents a major source of genomic data which can be used as an input to the framework rather than simulated data. This would alter the approach of the framework considerably. If case-control SNP genotype data was obtained from HUNT and used with the framework, then disease status would already have been determined, which would remove the need for a disease assignment step in the framework. Alternatively, in the context of planning case-control studies on HUNT data, disease status could be temporarily removed from the HUNT case-control data and the data processed in regular framework mode in order to provide pre-study assessments on the required sample size (number of cases and controls) needed in order to expect detection of a candidate SNP assuming a given disease model and association method. Another use in the context of HUNT data could be this: If other case-control studies on HUNT SNP data detected one or more SNPs that were later confirmed in laboratories to be causal for the disease, then the framework could be used on the same case-control data, removing disease status, then assuming the already detected SNP as the disease SNP and exploring the performances of a wide range of disease models for that SNP and a wide range of methods. In this way, the framework's components could provide methodic insights using real data with a known and confirmed disease SNP.

With this discussion of the test results of the framework, the characteristics of its current state and the potential for further applications, the thesis is concluded.

Bibliography

Alan Agresti. *Categorical Data Analysis, Second Edition.* Wiley InterScience, 2002.

P. Armitage. Tests for Linear Trends in Proportions and Frequencies. *Biometrics*, 11(3):375–386, 1955.

D. J. Balding, M. Bishop, and C. Cannings, editors. *Handbook of Statistical Genetics.* John Wiley & Sons, Ltd., West Sussex, England, 2003.

David J. Balding. A tutorial on statistical methods for population association studies. *Nature Reviews Genetics*, 7(10):781–791, 2006.

Yoav Benjamini and Yosef Hochberg. Controlling the False Discovery Rate: a Practical and Powerful Approach to Multiple Testing. *J. Roy. Statist. Soc.*, 57 (1):289–300, 1995.

N.E. Breslow and N.E. Day. *Statistical methods in cancer research vol. 1: The analysis of case-control studies.* International Agency for Research on Cancer, 1980.

A. Malcolm Campbell and Laurie J. Heyer. *Discovering Genomics, Proteomics, & Bioinformatics, second edition.* Pearson Education, Inc., San Fransisco, CA, 2007.

William G. Cochran. Some Methods for Strengthening the Common #2 Tests. *Biometrics*, 10(4):417–451, 1954.

Heather J. Cordell. Epistasis: what it means, what it doesn't mean, and statistical methods to detect it in humans. *Human Molecular Genetics*, 11(20):2463–2468, 2002.

Heather J. Cordell and David G. Clayton. A Unified Stepwise Regression Procedure for Evaluationg the Relative Effects of Polymorphisms within a Gene Using Case/Control or Family Data: Application to HLA in Type 1 Diabetes. *Am. J. Hum. Genet.*, 70(1), 2002.

Heather J Cordell and David G Clayton. Genetic association studies. *The Lancet*, 366(9491):1121–31, 2005.

Annette J. Dobson. *An introduction to generalized linear models, second edition.* Chapman & Hall/CRC, 2002.

Scott M. Dudek, Alison A. Motsinger, Digna R. Velez, Scott M. Williams, and Marylyn D. Ritchie. Data Simulation Software for Whole-Genome Association and Other Studies in Human Genetics. In *Pacicic Symposium on Biocomputing*, volume 11, pages 499–510, 2006.

Sandrine Dudoit, Juliet Popper Schaffer, and Jennifer C. Boldrick. Multiple Hypothesis Testing in Microarray Experiments. *Statistical Science*, 19(1):71–103, 2003.

Boris Freidlin, Gang Zheng, Zhaohai Li, and Joseph L. Gastwirth. Trend Tests for Case-Control Studies of Genetic Markers: Power, Sample Size and Robustness. *Human Heredity*, 53(2):146–152, 2002.

Juan R Gonzalez, Lluis Armengol, Elisabet Guino, Xavier Sole, and Victor Moreno. http://cran.r-project.org/web/packages/SNPassoc/index.html, 2008.

David A. Hafler, Alastair Compston, Stephen Sawcer, Eric S. Lander, Mark J Daly, Philip L. De Jager, Paul I. W. de Bakker, Stacey B. Gabriel, Daniel B. Mirel, Adran J. Ivinson, Margaret A. Pericak-Vance, Simon G. Gregory, John D. Rioux, Jacob L. MacCauley, Jonathan L. Haines, Lisa F. Barcellos, Bruce Cree, Jorge R. Oksenberg, and Stephen L. Hauser. Risk Alleles for Multiple Sclerosis Identified by a Genomewide Study. *New England Journal of Medicine*, 357(9): 851–862, 2007.

Daniel L. Hartl and Elizabeth W. Jones. *Essential Genetics: A Genomics Perspective.* Jones and Bartlett Publishers, Sudbury, MA, 2006.

Andrew T Hattersley and Mark I McCarthy. What makes a good association study. *The Lancet*, 366(9493):1315–1323, 2005.

Joel N. Hirschhorn and Mark J. Daly. Genome-wide association studies for common diseases and complex traits. *Nature*, 6(2):95–108, 2005.

Josephine Hoh and Jurg Ott. Mathetical multi-locus approaches to localizing complex human trait genes. *Nature*, 4(9):701–709, 2003.

Arthur M. Lesk. *Introduction to Genomics.* Oxford University Press Inc., New York, 2007.

Lyle J Palmer and Lon R Cardon. Shaking the tree: mapping complex disease genes with linkage disequilibrium. *The Lancet*, 366(9492):1223–34, 2005.

Sandy B. Primrose and Richard M. Twyman. *Principles of Genome Analysis and Genomics, third edition*. Blackwell Science, Malden, MA, 2003.

Marita Risberg. Combining the max test with methods for family-wise error rate. Master's thesis, NTNU, 2008.

Ronald E. Walpole, Raymond H. Myers, Sharon L. Myers, and Keying Ye. *Probability & Statistics for Engineers and Scientists*. Pearson Education International, 2002.

Constance L. Wood. Comparison of Linear Trends in Binomial Proportions. *Biometrics*, 34(3):496–504, 1978.

Gang Zheng and Joseph L. Gastwirth. On estimation of the variance in Cochran-Armitage trend tests for genetic association using case-control studies. *Statist. Med.*, 25(18):3150–3159, 2006.

Appendix A

R-code

A.1 Complete R-code for framework

master.r

```
# master.r: skript for aa bruke rammeverkets metoder etter
    ferdig oppsett
#-------------------------------------------------------------

source("master/sourceskript.r")

set.seed(121)

datainfo <- skaffDataMedNcasesCtrls(ncases=100, nctrls=200,
    g=0.4, f0=0.3, f1=0.35, f2=0.35, kffiln="konfig.datasim
    ", utkat="utkatalog", dafiln="data.txt", egfiln=F)

data <- read.table(datainfo$datafil)

slemSNP <- finnBesteNslemSNP(data=data, g=datainfo$g, N=1)

outputobj <- testanalyse(data=data, n=100, slemSNP=slemSNP,
    penetr=datainfo$penetr)

frekvens <- blantBestePverdier(pverdier=outputobj[[2]],
    slemInd=slemSNP, besteN=5)

print(frekvens)
```

```
source("master/rmsourceskript.r")

print("ferdig")
```

sourceskript.r

```
# sourcescript.r: sourcescript for alle filene i koden
#-----------------------------------------

source("skaffDataMedNcasesCtrls.r")
source("skaffDataMedNcasesCtrls/finnNodvendigSampleSize.r")
source("skaffDataMedNcasesCtrls/lagGenomeSIMkonfigfil.r")
source("skaffDataMedNcasesCtrls/hentUtNUMGENS.r")
source("skaffDataMedNcasesCtrls/simulerDatasett.r")

source("finnBesteNslemSNP.r")
source("finnBesteNslemSNP/estimerGi.r")
source("finnBesteNslemSNP/hentUtBesteN.r")

source("testanalyse.r")
source("testanalyse/beregnSykdomskolonne.r")
source("testanalyse/lagKontingenstabell.r")

source("blantBestePverdier.r")
source("blantBestePverdier/blantBesteN.r")
```

rmsourceskript.r

```
# rmsourcescript.r: skript for aa rydde opp i R-workspacet
    etter bruk
#-----------------------------------

rm(skaffDataMedNcasesCtrls)
rm(finnNodvendigSampleSize)
rm(lagGenomeSIMkonfigfil)
rm(hentUtNUMGENS)
rm(simulerDatasett)
```

```
rm(finnBesteNslemSNP)
rm(estimerGi)
rm(hentUtBesteN)

rm(testanalyse)
rm(beregnSykdomskolonne)
rm(lagKontingenstabell)

rm(blantBestePverdier)
rm(blantBesteN)
```

skaffDataMedNcasesCtrls.r

```
# skaffDataMedNcasesCtrls.r: ...
#-----------------------------------

skaffDataMedNcasesCtrls <- function(ncases, nctrls, g, f0,
    f1, f2, kffiln, utkat, dafiln, egfiln=F) {

        utlist <- list()

        #-----------------------------------------------
        # Operasjon 1. Beregn nodvendig sample size for aa
            faa tilstrekkelig
        # mange cases og controls, gitt enkel 1SNP-
            penetransemodell
        #-----------------------------------------------

        ssize <- finnNodvendigSampleSize(ncases, nctrls, g,
            f0, f1, f2)
        tplist <- list(ssize)
        names(tplist) <- "ssize"
        utlist <- c(tplist, utlist)

        #-----------------------------------------------
        # Operasjon 2. Simuler datasettet i genomeSIM ved
            hjelp av enten
        # a) en kopi av en default-konfigfil som hentes fra
            et sted
        # i kildekoden, eller
```

```
# b) en helt egen konfigfil, som maa skaffes av
    brukeren selv
# og plasseres paa rotnivaaet i kildekoden for
    master.r kalles.
# Uansett hvilken metode som brukes vil den sample
    size som ble
# funnet over editeres automatisk inn i konfigfilen
    for simuleringen
# starter.
#-----------------------------------------------

konfig <- ""
if (egfiln) { # default=F
        konfig <- egfiln
} else {
        konfig <- lagGenomeSIMkonfigfil(filnvn=
            kffiln)
}

# Riktig sample size editeres deretter automatisk
    inn i konfigfil vha sed.
# Teknisk ad hoc-triks som kan vaere vanskelig aa
    forstaa motivasjonen for uten forklaring:
# as.integer(ssize)+1, som er POPSIZE, synes aa bli
    ganget med 10, noe som
# kan virke rart da POPSIZE faar en ekstra 0.
    AArsaken til at den blir ganget
# med 10 er en ad hoc-losning paa et problem som
    oppstod da det trengtes en
# ekstra bokstav paa slutten av POPSIZE-linja i
    tekstfila fordi sed
# fjernet den siste bokstaven paa hver eneste linje
    . 0'en fjernes altsaa, og
# POPSIZE blir staaende igjen riktig.

system(paste("sed '/^POPSIZE/c POPSIZE ", (as.
    integer(ssize)+1)*10, "' ", konfig, " >
    tempkonfig", sep=""))
```

```
# Kommandoen over skapte trobbel med et ekstra LF-
    linjeskift paa slutten
# av hver linje. GenomeSIM vil dessuten ha
    konfigfila med CRLF-linjeskift.
# De ekstra LF-linjeskiftene fjernes, og CRLF
    sikres,
# vha folgende to kommandoer:

system("sed 's/.$//' tempkonfig > tempkonfig2")
system("sed -e 's/$/\r/' tempkonfig2 > tempkonfig")

system(paste("cp tempkonfig", konfig)) # dette blir
    den endelige konfigfila
system("rm tempkonfig tempkonfig2") # rydder opp

# Simulerer datasettet. Det havner som en tekstfil
    i den
# spesifiserte utkatalogen. (En kopi av konfigfila
    som ble brukt
# havner ogsa/ der, som dokumentasjon for
    simuleringen.)

dtfiln <- simulerDatasett(kffiln=konfig, utkat=
    utkat, dafiln=dafiln)

# Setter opp et R-objekt som inneholder informasjon
# om datasettet som nettopp ble simulert.

tplist <- list(konfig)
names(tplist) <- "konfig"
utlist <- c(utlist, tplist)

tplist <- list(dtfiln)
names(tplist) <- "datafil"
utlist <- c(utlist, tplist)

tplist <- list(g)
names(tplist) <- "g"
utlist <- c(utlist, tplist)
```

```
        tplist <- list(c(f0,f1,f2))
        names(tplist) <- "penetr"
        utlist <- c(utlist, tplist)

        utlist
}
```

finnNodvendigSampleSize.r

```
# finnNodvendigSampleSize.r: ...
#-------------------------------

finnNodvendigSampleSize <- function(ncases, nctrls, g, f0,
    f1, f2) {
        g0 <- (1-g)^2
        g1 <- 2*g*(1-g)
        g2 <- g^2
        K <- g0*f0+g1*f1+g2*f2
        utsize <- max(ncases/K, nctrls/(1-K))
        utsize
}
```

lagGenomeSIMkonfigfil.r

```
# lagGenomeSIMkonfigfil.r: ...
#-----------------------------

lagGenomeSIMkonfigfil <- function(filnvn="konfig.datasim")
    {
        system(paste("cp skaffDataMedNcasesCtrls/
            genomeSIMfiler/genomeSIMkonfigfilMal.datasim",
            filnvn))

        filnvn
}
```

simulerDatasett.r

```
# simulerDatasett.r: ...
#-------------------------

simulerDatasett <- function(kffiln, utkat, dafiln) {

        # Forberedelse: lager utkatalog for simulerte filer
            , og
        # filene som genomeSIM lager kommer til aa faa
            forstavelsen "out"

        system(paste("mkdir", utkat))
        out <- paste(utkat, "/out", sep="") #

        system(paste("genomeSIM", kffiln, out)) # dette er
            simuleringskallet i Unix

        # En kopi av konfigfila fra rotnivaa i kildekoden
            legges i utkatalogen
        # som dokumentasjon paa simuleringen som ble gjort

        system(paste("cp", kffiln, utkat))

        # Trenger aa vite NUMGENS-tallet fra en linje i
            konfigfila for aa kunne
        # gi ut filbanen (adressen) til den ferdige
            datasettfila.
        # Skreller ogsaa vekk den forste kolonnen i
            datasettet ved hjelp av
        # et perl-kall. Kolonnen som fjernes er en
        # irrelevant genomeSIM-sykdomskolonne da kun
            tilfeldig sykdoms-
        # paalegging ble brukt i denne simuleringen. Det
            resterende datasettet
        # inneholder kun kolonner for simulerte genotyper.

        NUMGENS <- hentUtNUMGENS(kffiln)
        datfn1 <- paste(utkat, "/", "out.1.", NUMGENS, ".
            out", sep="")
        datfn2 <- paste(utkat, "/", dafiln, sep="")
```

```
        perlkall <- paste("perl -pe 's/^[^ ]+\ *//'",
            datfn1, ">", datfn2)
        system(perlkall)

        # gir ut filbanen (adressen) til datasettfila

        datfn2
}
```

hentutNUMGENS.r

```
# hentUtNUMGENS.r:
#-------------------

hentUtNUMGENS <- function(filnavn) {
        NUMGENS <- system(paste("grep -v \\#", filnavn, "|
            grep NUMGENS"), intern=T)
        NUMGENS <- strsplit(NUMGENS, " ")
        NUMGENS <- NUMGENS[[1]][2]
        NUMGENS <- strsplit(NUMGENS, "")
        NUMGENS <- NUMGENS[[1]]
        NUMGENS <- NUMGENS[1:(length(NUMGENS)-1)]
        NUMGENS <- paste(NUMGENS, collapse="")
        NUMGENS <- as.numeric(NUMGENS)
        NUMGENS
}
```

finnBesteNslemSNP.r

```
# finnSlemSNP.r: ...
#-------------------

finnBesteNslemSNP <- function(data, g, N) {

        # setter opp matrise som lagrer allelfrekvensdata
            for alle m SNP

        m <- dim(data)[2]
        mat <- matrix(nrow=m, ncol=6)
```

```
# fyller ut de 3 forste kolonnene i matrisen med
    estimerte
# frekvenser g0, g1 og g2 for alle m SNPx, ved
    hjelp av
# hjelpefunksjonen estimerGi.r
# (g0+g1+g2=1 for alle SNP)

mdata <- as.matrix(data)
g0hat <- apply(mdata, 2, estimerGi, i=0)
g1hat <- apply(mdata, 2, estimerGi, i=1)
g2hat <- apply(mdata, 2, estimerGi, i=2)
mat[,1] <- g0hat
mat[,2] <- g1hat
mat[,3] <- g2hat

# antar Hardy-Weinberglikevekt og beregner 3
    forskjellige
# estimater for g basert paa hhv g0, g1 og 2.
    Fyller dette
# inn i de 3 siste kolonnene i tabellen.

ghat0 <- 1-sqrt(g0hat)
ghat1 <- 1/2 - 1/2*sqrt(1-2*g1hat)
ghat2 <- sqrt(g2hat)
mat[,4] <- ghat0
mat[,5] <- ghat1
mat[,6] <- ghat2

dmat <- data.frame(mat)
names(dmat) <- c("g0hat", "g1hat", "g2hat", "ghat0"
    , "ghat1", "ghat2")

# Sammenligner den onskede verdien g med
    gjennomsnittet av de
# tre estimatene for hver SNP. Identifiserer en
    liste over de
# N SNP som er naermest den onskede g i folge dette
     kriteriet.
# Til dette brukes hjelpemetoden hentUtBesteN.r
```

```
# (Andre kriterier/normer kunne ogsaa vaert tenkt
    brukt)

# NB! Estimatet for g basert paa g1-kolonnen (
    kolonne 2 i matrisen)
# produserer iblant NA-verdier pga. kvadratroten av
    noe negativt.
# Dette
# skjer naar den opprinnelige
    simuleringsparameteren for en SNP
# ligger naer g=0.5. Programmet mitt haandterer
    dette ved aa se bort
# fra alle SNP som gir NA-verdier i kolonne 2.
# Dette betyr at hvis man er paa utkikk etter
# en SNP med en g-verdi i omraadet rundt 0.5, vil
    man risikere aa se
# bort fra enkelte gode kandidater.

# Det anbefales uansett at man i etterkant sjekker
    at de SNP
# i listen som kommer
# ut faktisk ligger "naer" den onskede verdien for
    g.

diff <- abs(g - 1/3*(ghat0+ghat1+ghat2))
diff[diff=="NaN"] <- 2
diff <- sort(diff)
diff[diff==2] <- "NaN"
utlist <- list()
tplist <- list(diff)
names(tplist) <- "besteSNP"
utlist <- c(utlist, tplist)
tplist <- list(dmat)
names(tplist) <- "Est_g012_g"
utlist <- c(utlist, tplist)

# Henter ut indeksene til de N SNP som har estimert
    g-verdi
# som ligger naermest den onskede g (ifolge det
    brukte naerhetsmaalet)
```

```
        # fra den sorterte differanselisten.
        # Eksempel paa output [N=5]: besteN = c
            (45,3,9,24,55)
        # Dette betyr at SNP nr. 45 har estimert g-verdi
            som ligger
        # naerest den onskede g som ble sendt inn til
            metoden, etterfulgt av
        # SNP nr. 3, 9, 24 og 55 i synkende rekkefolge.

        besteN <- hentUtBesteN(utlist$besteSNP, N)
        besteN
}
```

estimerGi.r

```
# estimerGI.r: ...
#----------------------

estimerGi <- function(SNPkolonne, i) {
        giHat <- length(SNPkolonne[SNPkolonne==i])/length(
            SNPkolonne)
        giHat
}
```

hentUtBesteN.r

```
# hentUtBesteN.r: ...
#----------------------

# Denne koden henter ut de N forste numeric-tall fra en
# character-vektor paa formen
# c("VX", "VY", ...) der X, Y, ... er tall.
# Eksempel: c("V14", "V6", "V107") og N=2, da gir koden ut
    c(14, 6).

hentUtBesteN <- function(besteliste, N) {
        besteliste <- names(besteliste[1:N])
        besteliste <- strsplit(besteliste, split="V")
        utindex <- vector(length=length(besteliste))
```

```
        for (i in 1:length(besteliste)) {
                utindex[i] <- as.numeric(besteliste[[i
                    ]][2])
        }
        utindex
}
```

testanalyse.r

```
# testanalyse.r
#-----------------

# Denne metoden gjor folgende n ganger:
        # paa datasett lastet i R, trill terning for
            sykdomspaaleggelse => produserer 1
            sykdomskolonne
        # tester en metode 1 (e.g. Armitage dominant) paa
            datasetet med tilhorende sykdomskolonne
        # tester en metode 2 (e.g. Armitage additiv) paa
            datasettet med samme tilhorende sykdomskolonne
        # ...
        # tester en metode M (e.g. Armitage recessiv) paa
            datasettet med samme tilhorende sykdomskolonne

# For hver av de n repetisjonene blir en ny sykdomskolonne
    produsert og de M metodene blir testet paa det samme
    opprinnelige datasettet, men naa med den nye
    sykdomskolonnen.

# La m vaere antall SNP. Outputobjektet er en liste:
        # liste[[1]] er n sett med sykdomskolonner (en
            POPSIZE x n matrise)
        # liste[[2]] er n sett vektorer med p-verdier for
            alle de m SNP'ene som stammet fra aa testet med
            metode 1 (en m x n matrise)
        # liste[[3]] er tilsvarende som liste[[2]], bare
            med metode 2 (en m x n matrise)
        # ...
        # liste[[M+1]] er tilsvarende som liste[[2]] og
            liste[[3]], bare med metode M (en m x n matrise)
```

```
# NB: Hvis antall metoder (M) skal okes utover 1, som er
    defaulten,
# maa dette editeres inn for haand i koden under. En slik
    utvidelse
# er stottet av
# kodedesignet, men gjores paa eget ansvar og kan medfore
    at koden ikke

# virker hvis det gjores feil.

#-----------------------------------

testanalyse <- function(data, n, slemSNP, penetr) {
        library(SNPassoc)

        M <- 1 # sett inn antall metoder som skal brukes
            her

        # Sett inn en vektor av metodenavn her.
        # Eksempel: c("metode1", "metode2", ...)
        # Dette er til bruk for navngiving av outputobjekt
            lenger nede
        # i koden.

        metnvn <- "max"

        # Setter opp et outputobjekt som bestaar av
            folgende:
        # En POPSIZE X n matrise med n sykdomspaalegginger
        # M matriser, hver av dimensjon m X n, med
        # n p-verdikolonner fra bruk av de M metodene

        POPSIZE <- dim(data)[1]
        m <- dim(data)[2]
        utlist <- list()
        tplist <- list(matrix(nrow=POPSIZE, ncol=n))
        names(tplist) <- "sykdomskol"
        utlist <- tplist
        for (i in 1:M) {
```

```
            tplist <- list(matrix(nrow=m, ncol=n))
            names(tplist) <- metnvn[i]
            utlist <- c(utlist, tplist)
        }

        # Her kjores for-lokken n ganger, og de nevnte
            outputmatriser
        # fylles ut. Metoden som lager p-verdiene i default
            -innstillingen
        # er maxstat fra SNPassoc-pakken i R.
        # maxstat trenger inputdata for 1 SNP paa formen

        # r0 r1 r2
        # s0 s1 s2 (se Zheng og Gastwirth, 2006)

        # Dette skaffes til veie av hjelpemetoden
            lagKontingenstabell.r

        for (i in 1:n) {
            utlist$sykdomskol[,i] <- as.matrix(
                beregnSykdomskolonne(data[slemSNP],
                penetr))
            for (j in 1:m) {
                table23 <- lagKontingenstabell(as.
                    matrix(data[,j]), as.matrix(
                    utlist$sykdomskol[,i]))
                max <- maxstat(table23)
                utlist[[2]][j,i] <- max[5]
            }
        }

        # Metoden gir per default matrisen med p-verdier
            fra maxstat

        utlist
}
```

beregnSykdomskolonne.r

```
# beregnSykdomskolonne.r - ...
```

```
#-----------------------------------

beregnSykdomskolonne <- function(SNPkol, penetr) {
        f0 <- penetr[1]
        f1 <- penetr[2]
        f2 <- penetr[3]
        utkol <- SNPkol
        utkol[utkol==0] <- rbinom(length(utkol[utkol==0]),
            1, f0)*3
        utkol[utkol==1] <- rbinom(length(utkol[utkol==1]),
            1, f1)
        utkol[utkol==2] <- rbinom(length(utkol[utkol==2]),
            1, f2)
        utkol[utkol==3] <- 1
        utkol
}
```

lagKontingenstabell.r

```
# lagKontingenstabell.r: ...
#-----------------------------------

# Denne metoden tar inn et datasett med SNP-genotyper og en
# tilsvarende sykdomskolonne. Fra dette teller den opp de
# relevante storrelser og setter opp en 2 X 3
    konfidenstabell
# paa formen

# r0 r1 r2
# s0 s1 s2

# Se Zheng og Ghastwirth, 2006

lagKontingenstabell <- function(SNPkol, sykkol) {
        mat <- cbind(SNPkol, sykkol)
        temp <- mat[mat[,1]==0]
        r0 <- length(temp[temp==1])
        mat[,1][mat[,1]==1] <-3
        temp <- mat[mat[,1]==3]
        r1 <- length(temp[temp==1])
```

```
        mat[,1][mat[,1]==3] <-1
        temp <- mat[mat[,1]==2]
        r2 <- length(temp[temp==1])
        n0 <- length(SNPkol[SNPkol==0])
        n1 <- length(SNPkol[SNPkol==1])
        n2 <- length(SNPkol[SNPkol==2])
        s0 <- n0-r0
        s1 <- n1-r1
        s2 <- n2-r2
        kottab <- rbind(c(r0,r1,r2),c(s0,s1,s2))
        kottab
}
```

blantBestePverdier.r

```
# blantBestePverdier.r: ...
#-------------------------

# Denne metoden tar inn en m X n tabell som bestaar av
# n sett med p-verdier for m SNP som stammer fra en p-verdi
  -
# genererende metode som er gjentatt n ganger paa alle de m
  SNP.
# slemInd er indeksplasseringen, f.eks 58, til Ã¨n spesiell
  SNP
# blant de m.
# besteN er et tall, f.eks 7. Metoden gir ut andelen av
  ganger
# den spesielle SNP'en er blant de laveste besteN p-
  verdiene
# for hver av de n gjentakelsene.
# Eksempel: slemInd = 58, besteN = 7, og anta at i p-
  verditabellen
# som kommer inn er n = 10.
# Da gir metoden ut en vektor med 10 frekvenser mellom 0 og
  1, der
# hver frekveks svarer til hvor ofte SNP nr 58 var blant de
  laveste
# 7 p-verdiene for den gjentakelsen av den p-
  verdigenererende
```

```
# metoden.

blantBestePverdier <- function(pverdier, slemInd, besteN) {
        n <- dim(pverdier)[2]
        m <- dim(pverdier)[1]
        vektor <- apply(pverdier, 2, blantBesteN, slemInd,
            besteN)
        gunstige <- length(vektor[vektor==1])
        mulige <- length(vektor)
        frekv <- gunstige/mulige
        print(vektor)
        frekv
}
```

blantBesteN.r

```
# blantBesteN.r:
#-----------------

# Denne metoden tar inn en p-verdikolonne (pvkol).
# Hvis p-verdi nummer slemInd er blant de N laveste i
# p-verdikolonnen, gir metoden ut tallet 1. Ellers gir den
# ut tallet 0.

blantBesteN <- function(pvkol, slemInd, N) {
        mat <- cbind(pvkol, 1:length(pvkol))
        mat <- mat[order(mat[,1]),]
        slemRank <- which(mat[,2]==slemInd)
        ut <- vector(length=1)
        if (slemRank > N) {
                ut <- 0
        } else {
                ut <- 1
        }
        ut
}
```

R-code for plotting

addomplot.r

```
# addomplot.r: skript for aa lage plottene for additiv og
    dominant modell

source("plott.r")

tall <- c(0,1,2)
penadd <- c(0.2,0.5,0.7)
pendom <- c(0.2,0.7,0.7)
penrec <- c(0.2,0.2,0.7)
penodm <- c(0.2,0.7,0.2)

par(mfrow=c(2,2))
plott(tall,penadd,main="Additive model")
plott(tall,pendom,main="Dominant model")
plott(tall,penrec,main="Recessive model")
plott(tall,penodm,main="Overdominant model")
dev.copy2eps(file="modelplots.eps")
```

plott.r

```
plott <- function(x,y,main) {
        plot(x,y,xlab="Number of M alleles", ylab="
            Penetrance (f)", pch=20, main=main, xaxp=c
            (0,2,2), yaxp=c(0,1,5), xlim=c(0,2), ylim=c(0,1)
            , cex=2, cex.axis=2, cex.lab=1.6, cex.main=2)
}
```

resultat6.r

```
par(mfrow=c(3,2))
resultat(outputobj, slemSNP, "p-values of disease SNP", "p-
    value rank of disease SNP")
resultat(outputobj, slemSNP+1, "p-values of random SNP", "p
    -value rank of random SNP")
```

```
resultat(outputobj, slemSNP-1, "p-values of random SNP", "p
    -value rank of random SNP")
dev.copy2eps(file="100.eps")
```

resultat.r

```
resultat <- function(outputobj, slemSNP, mainhist, mainkum)
    {
        slemSNp <- outputobj[[2]][slemSNP,]
        hist(slemSNp, breaks=20, main=mainhist, xlab="p-
            value range", cex=2, cex.axis=2, cex.main=2, cex
            .lab=1.6, density=200, col="blue")
        source("master/sourceskript.r")
        n <- dim(outputobj[[2]])[1]
        x <- 1:n
        y <- numeric()
        for (i in 1:n) {
                y <- c(y, blantBestePverdier(outputobj
                    [[2]], slemSNP, i))
        }
        plot(x,y, type="s", main=mainkum, xlab="Lowest N p-
            values", ylab="Rate among", cex=2, cex.main=2,
            cex.axis=2, cex.lab=1.6, ylim=c(0,1), pch=20,
            yaxp=c(0,1,2), lwd=3)
        abline(0,1/100)
        y
}
```