An R package for analysis of whole-genome association studies

David Clayton and Hin-Tak Leung
Juvenile Diabetes Research Foundation/Wellcome Trust
Diabetes and Inflammation Laboratory,
Cambridge Institute for Medical Research,
University of Cambridge

Short title: R package for whole–genome association

Corresponding author:
David Clayton
Juvenile Diabetes Research Foundation/Wellcome Trust Diabetes and Inflammation Laboratory
Cambridge Institute for Medical Research
Addenbrooke’s Hospital
Cambridge CB2 0XY
United Kingdom
Tel: 44 (0)1223 762669
Fax: 44 (0)1223 331206

Keywords: Statistical genetics, Whole-genome scan, Association analysis

Abstract

Objective

To provide data classes and methods to facilitate the analysis of whole genome association studies in the R language for statistical computing.

Methods

We have implemented data classes in which each genotype call is stored as a single byte. At this density, data for single chromosomes derived from large studies and new high-throughput gene chip platforms can be handled in memory. We use the object–oriented programming model introduced with version 4 of the S-plus package, usually termed “S4 methods”.

Results

At the current state of development the package only supports population–based studies, although we would hope to provide support for family–based studies soon. Both quantitative and qualitative phenotypes may be analysed. Flexible association testing functions are provided which can carry out single SNP tests which control for potential confounding by quantitative and qualitative covariates. Tests involving several SNPs taken together as “tags” are also supported. Efficient calculation of pair-wise linkage disequilibrium measures is implemented and data input functions include a function which can download data directly from the international HapMap project website.

Introduction

Most of the computational tools for analysis of genetic data are implemented as "standalone" programs and the practical analyst must become adept at interfacing these programs via intermediate files. There are, however, considerable advantages in integrating such tools within a general-purpose statistical computing environment which offers high quality graphics and extensive data visualization tools. When we were about to embark on a very large scale whole-genome association study we decided to take the opportunity to attempt to design such a system. In this task we were aided by our experience of a whole–genome association study of some 13,000 non-synonymous SNPs in 4,000 cases and 4,000 controls [1], a study which has taught us the need to critically evaluate the data using the full armoury of data analytic tools.

Although several statistical systems were possible candidates, we chose the R language and environment for statistical computational. Strong points in its favour were

1. it has great flexibility of the data structures it can handle,
2. it supports object–oriented programming constructs,
3. it has a clearly defined "foreign language" interface, allowing inclusion of efficient modules coded in Fortran, C or C++,
4. there is an existing "genetics" package which offers many smaller scale tools,
5. it is widely used in bioinformatics due, in part, to the "Bioconductor" project (`http://www.bioconductor.org`),
6. finally, it is open–source software, distributed under the GNU public licence (`http://www.gnu.org/copyleft/gpl.html`).

However, there was one factor which argued against use of R, namely that R requires data to be held within RAM memory. Further, R, like its commercially distributed cousin S-plus, is profligate in its use of memory to store data objects. A real concern was, therefore, whether R could deal with the large volumes of data generated by whole–genome association studies. The current generation of whole–genome SNP chips yield approximately 500,000 SNP genotypes and association studies could involve as many as 10,000 subjects. Thus the data could extend up to 5×10^9 genotypes. However, little would be lost by storing the data by chromosome and this would reduce the maximum data object size by an order of magnitude. Further, R has a data type (termed "raw", and only fully implemented in versions 2.3 and later) which stores single byte variables. Use of this data type reduces the maximum data object size to 500 Mb — well within the capacity of modern computers and R's current maximum vector length ($2^{31} - 1 \approx 2 \times 10^9$. Since, a SNP genotype could theoretically be stored in 2 bits, even larger datasets could be stored by packing 4 genotypes to each single byte variable, but we judged this to be unnecessary at present.

Implementation of the `"snp.matrix"` class

The usual object for storing datasets in R is the *data frame*. This is convenient in that different types of variable (character, numeric, categorical "factors" etc.) may be held within the same object. However this carries a considerable housekeeping overhead. Instead we decided to store SNP data as a simple matrix, with storage mode "raw". Rows of the matrix correspond to samples (and, usually, to subjects) and columns correspond with different SNPs. The row names of each matrix will be subject (sample) identifiers which provide a link to the row names of a data frame containing the rest of the subject data, such as phenotype. We term this the "subject support " frame. Similarly the column names of the SNP matrix are SNP identifiers which link to a "SNP support " data frame, which contains information about each SNP such as chromosome position, RS number, and so on.

To facilitate use of such objects we employed the object-oriented programming features implemented in the R "methods" package. These are usually referred to as "S4 methods" since they were introduced in version 4 of the S-plus package. That implementation is described in the "green book" of Chambers[2]. The current R implementation differs from the original implementation in a number of ways, fully documented on the R developers web site (`http://developer.r-project.org/`). For the matrix holding SNP data, we have defined the class `"snp.matrix"`. This has storage mode raw, and inherits methods from the simple matrix class. Elements are coded zero, denoting missing data, or 1, 2 or 3 for the three possible genotypes (with homozygots coded 1 or 3 and heterozygots coded 2). For the X chromosome, we further defined the `"X.snp.matrix"` class. This inherits from the `"snp.matrix"` class, but has an additional slot containing the *sex* of each subject. Females are coded in the same way as for autosomal SNPs, and males are coded as if they were homozygous females. Finally we defined vector classes, `"snp"` and `"X.snp"` which hold single rows or columns of the corresponding matrix objects.

Several standard R functions are overloaded to behave correctly with the objects of the new classes. These include the sub-setting operator `[,]` which is used to extract portions of a matrix, the function `is.na()` which tests for missing values, and the functions `rbind()`, `cbind()` which join matrices to make larger matrices. The generic function `summary()` computes, for each SNP in the matrix, the call rate (proportion of non-missing values), the minor allele frequency, the genotype frequencies, and a z-test for Hardy-Weinberg equilibrium. These results are returned as a data frame. For speed, these computations (and those in most other functions requiring extensive numerical calculation) were programmed in C. Using a 2.2GHz Opteron 275 processor, it took 1.5 seconds to compute the summary of a test dataset with 13,549 autosomal SNPs measured in 7,276 subjects. A simple illustrative example is shown below

```
summ <- summary(chrom1)
chrom1a <- chrom1[, summ$MAF>0.01]
```

Here, `"chrom1"` is the original SNP matrix and a new matrix, `"chrom1a"`, is computed containing only those SNPs with minor allele frequency exceeding 1%.

For integration with the rest of the R system, it is also necessary to define "coercion" functions for conversion between the new classes of object and the conventional R classes. For S4 classes these functions take the form `"as(object, class)"`. We have defined such functions to coerce SNP matrix objects to simple numerical and character matrix types. We have also defined functions for coercion of simple matrices and data frames to SNP matrices although, because of the space requirements for these standard types, these functions will probably only find a use for small illustrative datasets. Finally, we provide coercion functions for objects of class `"snp"` and `"X.snp"`, representing single SNPs, to objects of class `"genotype"` in the R "genetics" package.

Statistical tests of association

At present we have only implemented functions for testing for genotype–phenotype association in population–based studies since the project which formed the impetus for this work is population–based. It is hoped that functions for family-based studies will be added at a later date.

The simplest and fastest function carries out 1 and 2 degree of freedom (df) tests for association between phenotype and SNPs, taken one at a time. The 1 df test is the usual Cochran-Armitage test and the 2 df test is the conventional Pearsonian chi-squared test for the 3×2 contingency table. An example of its use is

```
res <- single.snp.tests(cc, data=subject.data, snp.data=chrom1)
```

Here, "cc" is a case-control indicator found in the subject support frame "subject.data", and the function call will compute 1 and 2 df association tests for every SNP in the SNP matrix "chrom1". In our dataset of 13,549 SNPs in 7,276 subjects, these computations took 13.5 CPU seconds.

The "single.snp.tests" function will also compute "stratified" versions of these tests. These test for genotype–phenotype association within population strata defined by a third variable. The 1 df test is a generalization of the Cochran-Armitage test due to Mantel[3] and the 2 df test is based on the same principle *i.e.* calculating the "observed minus expected" scores and their variances *within* strata and then summing both score and variance across strata . In our hypothetical example, if the subject support frame also contained a categorical factor "region" giving geographical region of residence, we could protect against a possible confounding effect of different geographical distribution of cases and controls by using the command

```
res <- single.snp.tests(cc, region, data=subject.data, snp.data=chrom1)
```

This refinement adds scarcely at all to computing time. In our real example, the cases and controls were distributed across 12 geographical regions but the stratified tests took only 13.6 CPU seconds.

The remaining two functions for association testing are based on score tests in generalized linear models (GLMs)[4]. First, a "base" GLM, which forms the null hypothesis, is fitted. A test for additional effects of new terms in the model is then carried out by computing the "score" vector and its variance. The score vector is the first derivative of the log likelihood function with respect to the additional parameters to be introduced in the model and an estimate of its variance is obtained from the information matrix. In our implementation, the dependant variable may be assumed to have a distribution drawn from one of the binomial, Poisson, Gaussian or gamma families, and the "link" function which relates the expected value of the dependent variable to the linear model may be one of the logit, log, identity or inverse functions.If desired, a variance estimate which is robust against misspecification of the distribution family may be used in place of the model–based estimate[5]. This may be further extended in the manner usually described as the "Huber-White" approach [6] to allow for the case in which units are not mutually independent but fall into "clusters" of correlated observations.

In the first function, "snp.rhs.tests", we enter SNPs into a GLM on the right hand side *i.e.* as predictor variables, with the phenotype as dependent variable. We first fit a base GLM, and then test for addition of each SNP into this model. Only the "additive" component of the SNP is considered, so that each SNP is entered as a quantitative variable coded 0, 1 or 2, providing a 1 df test. In the simplest case, this is formally the same as the Cochran-Armitage test. For example, in our hypothetical example, these could be could be calculated by the command

```
crude <- snp.rhs.tests(cc ~ 1, family="binomial",
                        data=subject.data, snp.data=chrom1)
```

Here the argument "$cc \sim 1$" is a model formula indicating a base model containing only an intercept. The stratified tests could be calculated by

```
res <- snp.rhs.tests(cc ~ strata(region), family="binomial",
                        data=subject.data, snp.data=chrom1)
```

The use of the "strata" function in the base model formula is borrowed from the "survival" package in R. It allows the program to exploit a computational simplification which can be achieved if the GLM contains a stratification. This simplification is particularly dramatic if there are a large number of strata and if, as here, the model contains *only* a stratification. In this latter case, no iteration is then required to fit the base GLM by maximum likelihood. If, instead, we were to invoke the function as follows:

```
res <- snp.rhs.tests(cc ~ region, family="binomial",
                        data=subject.data, snp.data=chrom1)
```

where "region" is a factor on M levels, then region would enter the model as $M-1$ indicator variables and the computational effort in fitting the base model would be the same as fitting a logistic regression with $M-1$ covariates. Computation times in this case depend markedly on the proportion of genotype data that are missing. If there are no missing genotypes, the base GLM needs only to be fitted once and computation is very fast indeed. However, if genotypes are missing, the base GLM should be refitted each time, using only the subset of subjects with observed genotypes. In practise, we can skip this refitting if the proportion of missing genotype data is small. The default behaviour of the function is somewhat conservative, refitting if the proportion of missing genotypes exceeds 1%). In our real example, which contained a large amount of missing genotypes, calculation of the stratified tests took 21.7 seconds with use of the "strata" function and 258 seconds without.

The "snp.rhs.tests" function can also calculate tests for simultaneous inclusion of several "tag" SNPs in the phenotype model, the null hypothesis being that there is no association with any of the SNPs. These tests are closely related to Hotelling's T^2 statistics[9] the use of which has been proposed by several authors [10, 11, 12, 13]. They are specified by a further argument to the function, which provides a *list* of the sets of SNPs to be tested. Thus suppose we wish to to test the groups of SNPs in columns $\{1, 2, 3\}$, $\{4, 6\}$, $\{5, 7, 9\}$ etc., then this could be achieved by

```
taglist <- list(c(1,2,3), c(4,6), c(5, 7, 9), ...)
res <- snp.rhs.tests(cc ~ region, family="binomial",
                        data=subject.data, snp.data=chrom1, tests=taglist)
```

Currently tag SNPs are entered only as "main effects", so that the degrees of freedom for each test is equal to the number of SNPs in the group, unless there is very strong linear dependency between them, when one or more will be dropped. As we have argued elsewhere[13], this is optimal when linkage disequilibrium (LD), as measured by Lewontin's D', between SNPs within a group is close to one. If there are common recombinant haplotypes, however, power can be increased by addition of "interaction terms" and we hope to allow for this in future releases. Sets of SNPs such as illustrated by `"taglist"` in the above example will not usually be entered by hand in this manner, but sets likely to be informative if taken together will either be available from the process used in design of the genotyping chip, or can be calculated based on linkage disequilibrium measures. Tools for analysis of linkage disequilibrium in large-scale SNP data will be discussed in the next section.

The second function for GLM–based tests, `"snp.lhs.tests"`, as its name suggests, considers each SNP on the left hand side of a GLM *i.e.* as the dependent variable. Each autosomal SNP genotype is treated as a binomial variate with two "trials". The logit link function is used so that the GLM is formally the same as the logistic regression model. Note that this assumption implies Hardy-Weinberg equilibrium conditional upon any covariates fitted in the base model, but this assumption can be relaxed by use of the robust variance estimate. The "base" model and additional terms to be tested are specified using model formulae which omit the dependent variable. For the simplest case of an unstratified 1 df test similar to the Cochran-Armitage test, the use of the function would be as follows:

```
res <- snp.lhs.tests(chrom1, ~ 1, ~ cc, data=subject.data,
                          robust=TRUE)
```

This tests the effect of adding the case/control indicator to a base logistic model which takes each SNP genotype as dependent variable and includes only an intercept term. The equivalent stratified tests would be produced by

```
res <- snp.lhs.tests(chrom1, ~ strata(region), ~ cc,
                          data=subject.data, robust=TRUE)
```

or, at much greater computational expense, by

```
res <- snp.lhs.tests(chrom1, ~ region, ~ cc,
                          data=subject.data, robust=TRUE)
```

In our real example, the times taken for these commands were 21.3 and 280 seconds respectively.

A natural question is which of the two GLM tests is the "correct" one? It is well-known in epidemiology that, even when subjects are sampled according to disease status, as in case–control studies, a logistic regression model which treats disease status as the outcome variable reproduces identical estimates and variances for odds ratios for categorical exposures[7]. In this case, both directions of argument lead to identical conclusions. In general, both approaches are valid and the choice between them should be made on pragmatic grounds. For case–control studies, even though it seems unnatural, the logistic regression model with disease status as outcome is almost invariably preferred since it avoids complex or restrictive models for potentially multivariate risk factors. In our setting, pragmatic considerations should also apply. Treating phenotype as outcome automatically avoids the assumption of Hardy-Weinberg equilibrium for the distribution of genotype, and easily implements generalized multi–locus tests phenotype on the left hand side of the model formula is to carry out multi-locus tests. while, when SNP genotype is the outcome variable, avoidance of this assumption requires the use of "robust" variance estimates. the discussion of these issues in relation to sampling based on the values of quantitative phenotypes has been discussed by Wallace *et al.* [8].

Of course, it is better to carry out these crude and stratified 1 df tests using the simpler and faster "single.snp.tests" function. However, the GLM tests are more flexible. For example, they would allow adjustment for population substructure using continuous scores derived from principle components analysis[14]. Another example of this greater flexibility is that "snp.lhs.tests" could be used to test for allele frequency differences between geographical regions:

```
res <- snp.lhs.tests(chrom1, ~ strata(cc), ~ region,
                                  data=subject.data, robust=TRUE)
```

Linkage disequilibrium

Much has been written concerning calculation of measures of linkage disequilibrium between pairs of SNPs. The main computational problem is resolution of unknown haplotype phase. In a population-based study we observe a 3×3 contingency table of genotype counts, while we would like to have the 2×2 table of haplotype counts. This is usually approached by maximum likelihood estimation using the EM algorithm [15] but we use a different computational method, briefly outlined below.

The genotype and haplotype frequencies are illustrated in Table 1. It is the e subjects who are heterozygous at both loci that have unknown haplotype phase. Assuming Hardy–Weinberg equilibrium in the population, the multinomial probabilities for the 2×2 table of haplotype frequencies may be estimated by maximum likelihood and, in common with many "missing data" problems, at the solution the uncertain observations are distributed between possible states acording to expectation. Thus, if the expected proportion of the doubly heterozygous subjects to carry the haplotypes (u–v, U–V) is p, and $q = (1 - p)$ is the expected proportion to carry (u–V, U–v) then, at the maximum likelihood solution, the ratio p/q is equal to the odds ratio in the 2×2 table of haplotype frequencies. Thus we have

$$\begin{aligned}
A &= 2a + b + d + p.e \\
B &= 2c + b + f + q.e \\
C &= 2g + h + d + q.e \\
D &= 2i + h + f + p.e \\
p/q &= (A.D)/(B.C)
\end{aligned}$$

This leads to a cubic equation in p which may be solved directly using standard methods available in the GNU Scientific Library (`http://www.gnu.org/software/gsl/`). This is not only computational faster than the EM algorithm, which is iterative, but the presence of multiple roots is immediately obvious and it is trivial to select the root with the larger likelihood. Once we have solved for p and computed the table of haplotype frequencies, it is trivial to calculate any measure of disequilibrium, of which the most important are D' and r^2[16].

The R function which performs these calculations is "ld.snp". Its simplest use is

```
 res <- ld.snp(chrom1, depth = 100)
```

This calculates LD measures between each SNP and the 100 neighbours on either side of it. The result, here "res", is an object of class "snp.dprime"; this basically a list of four band–diagonal matrices, in compact storage mode, holding values of D', r^2, r, and LOD (the log likelihood ratio comparing the hypotheses of association and no associonation between loci). We envisage that the major use of these functions will be, in conjunction with data from the HapMap project[17], calculation of "tag" SNPs for defined regions, and identification of clusters of SNPs for use in multiple-SNP association tests. When applied to the HapMap data for 30 trios from the Centre d'Étude du Polymorphism Humain collection (CEU), treated as unrelated individuals, the calculation of LD measures for the 300,000 SNPs on chromosome 1 to depth 100 took 73 seconds.

For visualization, we provide the function "plot.snp.dprime", which displays a diagram similar to that available from the familiar Haploview software[18] (http://www.broad.mit.edu/mpg/haploview/) as an encapsulated postscript (eps) file. Optionally, this can include annotation which, after conversion of the eps file to portable document format (pdf), can be displayed by Acrobat Reader (http://www.adobe.com/products/acrobat/) (although not, currently, by other pdf viewers).

Extension of the computations to deal with data from nuclear families (*e.g.* case–parent trios) should not be difficult; one would use only parents (*i.e.* founders), for estimation of the two-SNP haplotype frequencies, but the phase of many of the *e* heterozygous genotypes of Table 1 will be resolvable given offspring genotype. For the remainder, the same method as outlined above should be satisfactory. However, this is not implemented at the time of writing.

Data input and output

Eventually the package will need to include data input routines for a number of input file layouts and formats. Currently we have three input routines. The most flexible is "read.snp.long". This reads a "long" data file in which each SNP call is on a single line, preceded by sample and snp identifiers. Each line can also contain a confidence score, allowing filtering of calls to be treated as valid.

Two commonly used "wide" formats are also supported. The function "read.snp.pedfile" reads "pedfiles" in which there is one line of data per subject, commencing with six standard fields describing relationship between subjects (if any), sex, and disease status, and followed by genotypes, coded as pairs of alleles. The third input routine reads "HapMap" style input files, in which there is one line of data for each SNP, commencing with some data about the SNP, and followed by the genotypes recorded for all the subjects in the collection. Since a major use of this function will be to read data from the HapMap project, and these data are still somewhat volatile, this function can download data direct from the HapMap web site (http://www.hapmap.org/). Thus, to download and read in the chromosome 1 data for the CEU trios:

```
folder <- "http://www.hapmap.org/genotypes/latest/fwd_strand/non-redundant"
file <- "genotypes_chr1_CEU_r21_nr_fwd.txt.gz"
ceu.1 <- read.HapMap.data(paste(folder, file, sep="/"))
```

(to avoid an inordinately long line, the folder and file names have been entered as two strings and "pasted" together). The result. `"ceu.1"`, will contain a list with two elements containing the SNP matrix and a SNP support data frame (a subject support file for the CEU trios would have to be downloaded separately). These data concern approximately 300,000 SNPs in 90 subjects; the resultant `"snp.matrix"` data object occupies about 40Mb.

The package contains one output function, `"write.snp.matrix"`, modelled closely on the standard R function, `"write.table"`, which writes a data frame as a text file. Currently genotypes are written as single numbers (0, 1, or 2); greater flexibility will be Incorporated as necessary. The main use of this function is to facilitate export of data to other programs.

Discussion

As is apparent from the above description, further work is necessary for this package to provide a truly comprehensive set of tools for the analysis of whole–genome association studies. Nevertheless, even in its present state it provides tools to do much of what is required, at least for population–based studies and new techniques can be added with little difficulty. The current version of the p[ackage, provisionally named `"snpMatrix"`, may be downloaded from the web site of the first author, (`http://www-gene.cimr.cam.ac.uk/clayton/`).

Acknowledgements

We thank Jason Cooper and Chris Wallace for helpful suggestions and for finding many bugs. David Clayton is a Juvenile Diabetes Research Foundation/Wellcome Trust Principal Research Fellow. Hin-Tak Leung is funded by the Wellcome Trust as part of its support for the Wellcome Trust Case Control Consortium.

We thank the publishers, Karger AG (Basel), for permission to distribute this original typescript. The published article may be consulted at
`http://content.karger.com/ProdukteDB/produkte.asp?typ=pdf&doi=101422`

References

[1] Clayton D G, Walker N M, Smyth1 D J, Pask R, Cooper J D, Maier1 L M, Smink L J, Lam A C, Ovington1 N R, Stevens H E, Nutland S, Howson J M M, Faham M, Moorhead M, Jones H B, Falkowski M, Hardenbol P, Willis T D, Todd J A: Population structure, differential bias and genomic control in a large-scale, case-control association study. Nature Genetics 2005;37:1243–1246.

[2] Chambers J M: Programming with Data. Springer, New York, 1998.

[3] Mantel N: Chi-square tests with one degree of freedom: extension of the Mantel–Haenszel procedure. Journal of the American Statistical Association 1963;58:690–700.

[4] Nelder J, Wedderburn R: Generalized linear models. Journal of the Royal Statistical Society Series A 1972;135:370–384.

[5] Boos D: On generalized score tests. The American Statistician 1992;4:327–333.

[6] Aerts M, Molenberghs G, Ryan L M, Geys H: Topics in Modelling of Clustered Data, volume 96 of *Monographs on Statistics and Applied Probability*. Chapman and Hall, London, 2002.

[7] Prentice R, Pyke R: Logistic disease incidence models and case–control studies. Biometrika 1979;66:403–411.

[8] Wallace C, Chapman J M, Clayton D G: Improved power offered by a score test for linkage disequilibrium mapping of quantitative trait loci by selective genotyping. American Journal of Human Genetics 2006;791:323–331.

[9] Hotelling H: The generalization of Student's ratio. Annals of Mathematical Statistics 1931;2:360–378.

[10] Xiong M, Zhao J, Boerwinkle E: Generalized t^2 test for genome association studies. American Journal of Human Genetics 2002;70:1257–1268.

[11] Chapman J M, Cooper J D, Todd J A, Clayton D G: Detecting disease associations due to linkage disequilibrium using haplotype tags: A class of tests and the determinants of statistical power. Human Heredity 2003;56:18–31.

[12] Fan R, Knapp M: Genome association studies of complex diseases by case-control designs. American Journal of Human Genetics 2003;72:850–868.

[13] Clayton D, Chapman J, Cooper J: The use of unphased multilocus genotype data in indirect association studies. Genetic Epidemiology 2004;27:415–428.

[14] Price A, Patterson N, Plenge R, Weinblatt M, Shadick N, D. R: Principal components analysis corrects for stratification in genome-wide association studies. Nature 2006; 38:904–909.

[15] Slatkin R, Escoffier L: Testing for linkage disequilibrium in genotypic data using the expectation-maximization algorithm. Heredity 1995;76:377–383.

[16] Devlin B, Risch N: A comparison of linkage disequilibrium measures for fine–scale mapping. Genomics 1995;29:311–322.

[17] The International HapMap Consortium: The International HapMap project. Nature December 2003;426:789–796.

[18] Barrett J C, Fry B, Maller J, Daly M J: Haploview: analysis and visualization of ld and haplotype maps. Bioinformatics 2004;21:263–265.

Tables and Figures

SNP	SNP V		
U	vv	Vv	VV
uu	a	b	c
Uu	d	e	f
UU	g	h	i

Genotype counts
$(a+b+c+d+e+f+g+h+i=N)$

SNP	SNP V	
U	v	V
u	A	B
U	C	D

Haplotype counts
$(A+B+C+D=2N)$

Table 1: Genotype and haplotype frequencies for two SNPs