29th November, 1995.

GAS Manual

Analysis Modules

2.0

Chapter 1 : Introduction

This is the analysis manual for the Genetic Analysis System version 2.0. Before using the modules described in this manual, you should be familiar with the contents of the companion document "General User Guide" which explains how to use *GAS* and the data input/output formats required.

1.1 Contents

1.2 Obtaining GAS

For information on the latest version of the *GAS* program, manuals, and example files connect via WWW to

`http://info.ac.ox.uk/~ayoung/gas.html`

GAS is currently available on the following systems: IBM-PC DOS, Vax VMS, DEC Ultrix, IBM AIX, DEC Alpha OSF/VMS, SUN Solaris, SGi IRIX and SunOS.

1.3 Legal Notice

The *GAS* program, manuals and associated files are not public domain and may not be modified without express permission of the author. However they may be used and copied freely provided that:

(1) this notice is distributed along with the program,

(2) no charge is made to other parties for services reliant upon *GAS* without express permission from the author,

(3) you agree that neither the author, nor any institution with which he has been associated, accepts responsibility for either the accuracy of results produced by *GAS* nor for the use to which they are put,

(4) the use of *GAS* is acknowledged in any publications incorporating results produced using the program.

With respect to clause (4), the following form of words is suggested:

> "data was analyzed using the GAS package version 2.0
>
> (c) Alan Young, Oxford University, 1993-95."

You should also include any relevant references given in the *GAS* manuals.

1.4 Style

Examples of input files are shown `in square font`. File-names are shown *in slanted roman font*. General parameters for functions are shown *in italics*. Note that *GAS* converts all input to lower-case unless it is enclosed in "quotation" marks.

1.5 Errors

During operation the *GAS* program may produce four types of message:

INFO	information, note appended to output file
WARNING	possible error, note appended to output file
ERROR	serious error, exit from current *GAS* module
FAILURE	fatal error, immediate exit from program

after *GAS* has completed, a copy of these messages can be found in the logfile (assuming you have requested one with the `set logfile` command).

If the program terminates with the message "INTERNAL ERROR" then an internal trap has been triggered and the program will need to be modified and re-compiled to run on your system. In this case please notify Alan Young via email at "ayoung@vax.ox.ac.uk" and include the program version, the computer system, and the actual input files that were being used when the failure occurred.

1.6 Examples

To supplement the manuals there are a series of demonstration examples included as computer files in the distribution package, and it is highly recommended that these are used in conjunction with the text. Output files ending in '.ps' contain Postscript graphics and must be viewed either on a screen previewer or by sending them to a Postscript compatible printer. If you're learning how to use *GAS* then a good method is to modify the example files to explore how the various optional parameters affect the type of analysis performed.

1.7 Help!

This manual covers operation of *GAS* only, and is not intended to form a statistics text. References have been included for most of the analytical routines, and these should be consulted for further information on their applicability. Please remember: *GAS* is designed to perform fast accurate analyses of genetic data, but the results will only be useful if an appropriate choice of routines has been made.

The file *gas.faq* contains answers to some of the commoner questions that users have asked about *GAS*. If you have a query, it is suggested you first post it to the bulletin board bionet.molbio.gene-linkage, and if no suitable response arrives then email:

ayoung@vax.ox.ac.uk

Chapter 2 : Parameters

This chapter discusses some of the parameters that are common to a large number of the routines. General parameters for functions are shown *in italics.* To use the *GAS* analysis routines you enter them in the following format:

call anal(*parameters* **);**

where '**anal**' is the name of the routine, and *parameters* is a list of one or more items to be used in the analysis. The descriptions of the function parameters use the convention:

$\odot$	this is compulsory
$\star$	this is optional
$\heartsuit$	postscript graphics available

The output from routines are written to the file specified by **set outfile**, or to *gas.out* if this has not been done. To obtain graphical output you must use **set psfile** before giving the **program;** command.

2.1 Locus

Most of the routines require the names of loci to analyze. These are entered after the **locus** keyword.

2.2 Theta recombination Values

Some of the routines require values for the recombination fractions between loci, while certain others can use them if available. These are entered after the **theta** keyword, and may be given in 3 formats:

2.2.1 No sex-difference

The most basic case is that in which the recombination fractions between markers are the same for both male and female chromosomes, and this is entered into the relevant *GAS* functions using the keyword **theta** followed by a list of the recombination fractions in the same order as any loci specified. For example, with six loci, five theta values are required and these are specified as

theta $\theta_1\ \ \theta_2\ \ \theta_3\ \ \theta_4\ \ \theta_5$

where each θ_i must be a positive number in the range 0 to $\frac{1}{2}$.

2.2.2 Constant sex-difference

It is possible to specify that the male and female recombination fractions differ by a fixed constant multiple. This is done by placing the letter '`c`' after the male recombination fractions, followed by the constant multiplier. Hence, for three loci with male recombinations 0.2 and 0.3, and female recombinations 0.24 and 0.36, the parameters should be

```
theta 0.2 0.3 c 1.2
```

2.2.3 Arbitrary sex-difference

If there is no simple relationship between male and female recombination values, both can be specified by separating them with the letter '`f`'. Thus for four loci, male recombinations 0.11 0.23 0.17 and female recombinations 0.15 0.19 0.22 (in the same order) enter the parameter list

```
theta 0.11 0.23 0.17 f 0.15 0.19 0.22
```

2.2.4 Interference

Interference occurs when the probability of cross-overs occurring between adjacent loci is not independent. The current version of *GAS* does not support this option.

2.3 Showraw

Many of the routines are able to produce extra output showing intermediate stages in their calculations. This is requested by adding the `showraw` parameter to the function brackets. (You can also try increasing the overall `verbosity` level.)

2.4 Signif

Most of the routines will mark a significant result with the symbol '< +' if a p-value or lodscore passes a particular threshold (the default size of which depends on the actual routine). The `signif` parameter may be used to change this threshold:

`signif` *threshold-value*

2.5 Mapping Functions

A mapping function is used to convert from recombination fractions to physical distances along a chromatid. Some of the routines are able to display their results in terms of physical distance using the **mapfunc** option:

mapfunc *name-of-map*

The map functions implemented in *GAS* are[1]

[1] carfal

[2] haldane

[3] kosambi

2.6 Help

Most of the analysis routines accept **help** as a parameter, and will respond by showing the full[2] list of parameters available with them. Thus

```
call sibdes( help );
```

will show all the parameters available with the **sibdes** routine.

[1] **carfal** is the Carter-Falconer map.

[2] Some parameters not mentioned in the manuals may be listed. These should not be used except by arrangment with myself – AY.

Chapter 3 : Statistical Analysis

The *GAS* program provides facilities to perform some basic statistical analyses[3] of the input data. The routines available allow the dependencies between members of the same family to be studied.

[3] See "Numerical Recipes in C" W.H. Press, B.P. Flannery, S.A. Teulosky & W.T. Vetterling, Cambridge (2nd Ed.) for background information on some of these.

3.1 ROUTINE : DISSECT

The `dissect` routine performs general statistical analysis on the input data, the format being:

`call dissect(` *options* `);`

Where the options are:

⋆ `pedigree`	analyze properties of whole dataset
⋆ `family`	examine individual families
⋆ `locus`	analyze loci singly and in pairs
♡	display of statistical results

3.2 Pedigree

The `pedigree` option gives data on the total number of active individuals in the pedigree and their family structures, numbers of offspring, matings and siblings. To perform this type of analysis, give the command:

`call dissect( pedigree );`

3.3 Family

The `family` option displays data on one or more families, which including their size and numbers of male/female members, crosses (ie. matings) and generations. The format is:

`call dissect( family` *family_name(s)* `);`

If no families are named, then every family in the pedigree is analysed sequentially.

3.4 Locus

The `locus` option analyses the data for a particular locus. The general format is:

```
call dissect( locus locus_name(s) );
```

To run the analysis for the locus **height** include the following line in your gasfile program:

```
call dissect( locus height );
```

The type of analysis performed depends on the locus (see below). If several loci of the same type are listed within the brackets, *GAS* performs a pairwise analysis to show correspondences between their distributions.

3.4.1 Affection Locus Algorithms

A variety of contingency tables are constructed showing the affection relations between relatives — in particular the correspondences between the affection statuses of parents and of their children. If two or more loci are listed then contingency tables are created showing their joint pair-wise affectedness distributions and giving the p-value for the Null Hypothesis that the statuses at the loci are independent.

3.4.2 Binary Locus Algorithms

The number of positive, negative and unknowns for each factor are displayed. There is no paired-locus analysis.

3.4.3 Named Locus Algorithms

The input gene frequencies are compared with observed population frequencies, the latter also being broken down into parent and child categories (parents being defined as those subjects with no children in the pedigree and children being defined as those with no parents). If two or more loci are listed then a table is constructed showing how often the alleles at each locus are present in the same subject (note that a fully typed individual will generate 4 entries in this table).

3.4.4 Quantitative Locus Algorithms

The mean, median, extrema, std.deviation and various correlations between relatives are calculated and p-values assigned. If two or more loci are present then the correlations between their values are computed.

Example 3.1

The gasfile *dis.gas* loads pedigree data from *dis.ped* and uses `dissect` to perform several types of analysis on it. The results are written to the file *dis.out* with postscript graphical output sent to the file *dis.ps*

Chapter 4 : Sib-Pair Analysis

These *GAS* modules perform sib-pair analyses on sets of loci to determine the degree of allele sharing between full siblings, and hence to indicate the chances of linkage between the loci. Both identity by descent[4] (IBD) and identity by state (IBS) methods are implemented.

4.1 Weights

Sib-ships in which there are more than two siblings may have a disproportionate effect[5] on the results of a sib-pair analysis, and various weighting strategies have been developed in an attempt to compensate for this.

4.1.1 Weighting Categorical Traits

Some of the categorical sib-pair tests used the `weight` parameter to compensate for multi-pair sibships. Using `strict` weights the sharing information from a family of n siblings by

$$2/n,$$

and using `hodge` weights it by a factor of

$$\frac{4}{3}\left(\frac{2n-3+(\frac{1}{2})^{n-1}}{n(n-1)}\right).$$

For instance, the command

```
call sibdes( locus dis1 mk1 weight strict );
```

performs a IBD sib-pair analyses for the affection locus `dis1` versus the marker locus `mk1`. Any multiple sibships are given a strict weighting as described above.

4.1.2 Weighting Quantitative Traits

The Haseman-Elston routines involve performing a least-squares fit through a set of points (one point being generated by each pair). The `dfweight` parameter compensates for multi-pair sibships by reducing the number of degrees of freedom used when the significance of the best-fit line is assessed.

[4] Ref. "The general purpose sib-pair linkage test", L.S. Penrose, Ann. Eugen. 18, 120-124 (1953).
[5] Ref. "The Information Contained in Multiple Sibling Pairs", S.E. Hodge, Genet.Epid. 1:109-122 (1984).

4.2 ROUTINE : SIBDES

This routine performs basic IBD analysis on sib-pairs categorized according to affection status.

`call sibdes( locus` *locus_names options* `);`

The routine lists the various types of matings, the degree of allele sharing between sibs in each (and parental source), the 2-1-0 t_2 and χ^2 scores and associated probabilities, together with the exact 1-0 binomial probabilities.

⊙**`locus`**	list the affection and marker loci to be analyzed
⋆ **`alltypes`**	show sharing for not-affected, concordant and discordant[6] pairs
⋆ **`halfsib`**	show common-parent sharing for half-siblings
⋆ **`weight`**	options are **`strict`** and **`hodge`**

4.2.1 Algorithm Notes

The IBD sharing is shown both in terms of a 2:1:0 distribution (using sibs in which both parents are fully informative (*GAS* defines a fully informative parent as one for which it is possible to tell exactly what happened to their alleles during mating) but not intercrosses, a 2:1:0 distribution using the expected sharing all pairs which are at least partly informative, and a 1:0 distribution for each parental sex. Intercrosses are treated differently from the fully informative pairs to avoid biasing the results towards 1-sharing, as would happen if the intercrosses in which pairs definitely shared 2, 1 or 0 alleles were included but those in which sharing was undecidable between 2 and 0 were excluded. Note that the χ^2 probability is 2-sided, whereas the others are 1-sided.

If you have allele data for more than one named locus on a chromosome then, provided the recombination fraction between adjacent loci is less than 0.3, you will benefit from using the interval map version (**`sibides`**) of this routine.

4.3 ROUTINE : SIBMLS

This routine calculates the maximum-likelihood 2-1-0 IBD sharing distribution of markers (ie. named loci). In addition to the data used by **`sibdes`** it also utilizes partial information from cases in which the sharing cannot be unambiguously determined.

`call sibmls( locus` *locus_names options* **`);`**

⊙**`locus`**	list the affection and marker loci to be analyzed
⋆ **`alltypes`**	show non-affected and concordant pairs

4.3.1 Algorithm Notes

The maximum-likelihood estimate is restricted so that $2z_2 > z_1$ and $z_1 > 2z_0$ (the 'possible triangle' method[6]), where z_i is the fraction of siblings sharing i alleles IBD. The maximum is located in a two-phase search, using simulated annealing to explore the function range, then Brent's algorithm to refine converge about the highest point found.

[6] Ref. "Asymptotic Properties of affected sib-pair linkage analysis", P. Holmans, Am.Jou.Hum.Gen. 52:362-374 (1993).

4.4 ROUTINE : SIB2MLS

This routine calculates the joint maximum-likelihood 2-1-0 IBD sharing distribution of pairs across two named loci simultaneously.[7] In addition to the data used by **`sibdes`** it also utilizes partial information from cases in which the sharing cannot be unambiguously determined.

`call sib2mls( locus` *locus_names options* **`);`**

⊙**`locus`**	list the affection and marker loci to be analyzed
⋆ **`alltypes`**	show non-affected and concordant pairs
⋆ **`compare`**	show mls sharing for alternative sub-models
⋆ **`showraw`**	display raw sharing data

4.4.1 Algorithm Notes

The region of maximization is restricted according to the type of model being considered. The present version of *GAS* compares mls values for the single-locus, multiplicative and general models.

The maximum is located in a two-phase search, using simulated annealing to explore the function domain, then Powell's algorithm (using Brent for the 1-dimensional sub-stages) to refine converge about the highest point found.

[7] Ref. "Two-Locus Maximum LodScore Analysis of a Multifactorial Trait: Joint Consideration of IDDM2 and IDDM4 with IDDM1 in Type I Diabetes", H.J. Cordell et.al, Am.J.Hum.Genet 57:920-934 (1995).

4.5 ROUTINE : SIBSTATE

The **sibstate** routine performs Identity By State analysis on sib-pair data.

call sibstate(locus *locus_names options* **);**

⊙**locus**	list the affection and marker loci to be analyzed
⋆ **alltypes**	show non-affected and concordant pairs
⋆ **showraw**	display raw statistical information
⋆ **weight**	options are **strict** and **hodge**

For instance, the command

```
call sibstate( locus dis1 mk1 );
```

performs an IBS sib-pair analyses for the affection locus `dis1` versus the marker locus `mk1`.

4.5.1 Algorithm Notes

For the **sibstate** analysis (and any other IBS technique) it is absolutely essential that the allele frequencies of the marker loci are correctly set, otherwise the computed probabilities will be meaningless. This means that global binning (preferably using fixed bin sizes) must be used if data is read using the **alsize** option.

Note that even if parental information is available on some of the pairs, it will not be used in the analyses.

Two methods are used to calculate p-values. The first[8] uses a 2-sided χ^2 test to compare the overall observed IBS sharing distribution with that predicted from the allele frequencies — note that this can produce spurious significant results when an excess of 0 sharers are present. The second method[9] uses Lange's Z-statistic (which automatically takes into account multiple sibships) and produces a 1-sided p-value.

Note that the **weight** parameter only affects the χ^2 results by reducing the effective contribution of multiple sibships — the Z-statistic does not require weighting.

[8] Ref. "The Affected Sib-pair Method using Identity by State Relations", K. Lange, Am.Jou. Hum.Genet., 148-150 (1986).
[9] Ref. "A Test Statistic for the Affected Sib-set Method", K. Lange, Ann.Hum.Genet., 50 283-290 (1986).

4.6 ROUTINE : SIBMAP

This routine gives a graphical display of how sharing between siblings varies along the length of a chromosome, with options to estimate recombination fraction and named-locus order. The syntax is:

call sibmap(locus *locus_names options* **);**

⊙**locus**	list affection and marker loci to analyze
⋆ **bestorder**	attempt to order loci using sharing data
⋆ **halfsib**	show half-siblings with paternal/maternal options
⋆ **mapfunc**	select map-function for distance estimates
⋆ **maternal**	show map for maternally-derived chromosomes
⋆ **maxprob**	show pairs having crossover probability above this threshold
⋆ **minchanges**	show pairs in which there are at least n changes of sharing
⋆ **mindefinite**	show pairs in which at least n loci can be categorised definitely
⋆ **paternal**	show map for paternally-derived chromosomes
⋆ **sortleft**	sort pairs by first recombination position from left
⋆ **sortright**	sort pairs by first recombination position from right
⋆ **theta**	recombination values to use with maxprob

For instance, the command

```
call sibmap( locus dis1 dis2 mk1 mk2 mk3 mk4 maternal sortleft );
```

gives a map of the allele sharing in maternally-derived chromosomes for the marker loci `mk1`, `mk2`, `mk3` and `mk4`, sorted to show pairs with the most sharing at the left end of the chromosome first.

4.6.1 Algorithm Notes

The problem of computing a metric for all possible arrangements of loci is called N-P complete, meaning that the time required is proportional to the factorial of the number of possible orders. For modest numbers (above 8) calculating all possible orders is computationally impractical, and instead *GAS* uses a version of the Metropolis algorithm based on simulated annealing. Once a sufficiently good order has been produced, all sets of 3 adjacent loci are permuted to indicate regions where the order is least certain.

`bestorder` can take a numeric parameter n in which case the first n equivalently good orders (as produced by the triplet permutations described above) are listed in full.

Example 4.1

The gasfile *sib.gas* reads g-format locus data from *sib.loc*, and g-format pedigree data from *sib.ped*. The `sibdes` routine is used to perform an IBD analysis, with results sent to *sibp.out*. The `sibstate` routine is used to perform an IBS analysis, with results sent to *sibs.out*. The `sibmap` routine is used twice, firstly to show the sharing of paternal chromosomes for the sib-pairs in which at least three of the named markers are unambiguously determined (results in *sibm1.out*), and then to to show only those pairs in which there are at least two changes in sharing status[10] (results in *sibm2.out*).

Example 4.2

The gasfile *bo.gas* reads g-format locus data from *bo.loc*, and g-format pedigree data from *bo.ped*. The `sibmap` routine is used to determine the most probable order of the named loci 1-10 (which should be 1,2,3,...,10) with the results being written out to the file *bo.out*

The dataset contains 20 nuclear families with each locus simulated as having 6 equally frequent alleles and a recombination fraction of 0.04 between adjacent loci. The results show that there are two equally good orders in which loci 8 and 9 are interchanged (20 sib-pairs is too small a dataset to expect sufficient crossovers between each locus to produce a unique best ordering).

[10] This may be used to indicate the possibility of double recombinants.

4.7 ROUTINE : SIBMWU

The **sibmwu** routine performs a non-parametric IBD analysis on a trait which is specified in terms of a quantitative locus.

call sibmwu(locus *locus_names options* **);**

⊙**locus**	list the affection and marker loci to be analyzed
⋆ **exact**	sizes of dataset below which p-values calculated exactly
⋆ **signif**	significance level for linkage

For instance, the command

```
call sibmwu( locus humour mker );
```

assesses whether sibling pairs sharing more alleles at named locus 'mker' are significantly more similar at quantitative locus 'humour' than pairs sharing fewer alleles at 'mker'.

4.7.1 Algorithm Notes

The **sibmwu** routine first ranks all sibling pairs according to the absolute difference in their value at a quantitative locus, then uses the Mann-Witney U-test to compare the distributions of these values within subsets of the sib-pair population, categorized according to the amount of IBD sharing at a named locus. A result may indicate linkage if the average rank of pairs decreases as the number of alleles shared IBD increases, and the p-values are 1-sided towards this direction.

The **exact** parameter controls the threshold above which the U-statistic is calculated approximately. For more details see the entry on the **assmwu** routine.

4.8 ROUTINE : SIBHE

This routine implements the Elston-Haseman algorithm[11] for analyzing a quantitative trait using IBD sib-pair information.

`call sibhe( locus` *locus_names options* `);`

⊙`locus`	list the affection and marker loci to be analyzed
⋆ `absolute`	use absolute difference of values rather than square
⋆ `dfweight`	compensate for multi-pair sibships
⋆ `graph`	draw graphs of regression plots
⋆ `knownonly`	only pairs with unambiguous sharing are used
⋆ `showallp`	show p-values for +ve slope regressions
⋆ `signif`	the value at which significant results are marked
⋆ `useall`	pairs with no genetic sharing information are used
♡	regression plots with graph option

4.8.1 Algorithm Notes

The basic assumption of this method is that siblings sharing marker alleles near the quantitative trait locus will be more likely to have similar quantitative values than non-sharing siblings. Thus the mean value of the difference between siblings should decrease as the fraction of alleles shared increases. The `sibhe` routine performs a least-squares fit using allele sharing as the independent variable, and trait difference as the dependent variable. A significantly negative slope may be taken to indicate linkage.

`sibhe` implements 3 versions of the Haseman-Elston algorithm. The default is to use all pairs for which there is definite sharing information for either the paternal or maternal alleles (or their sum). The `knownonly` parameter causes *GAS* to use only the pairs for which there is definite sharing information for both paternal and maternal alleles.[12] The `useall` parameter means that all pairs in a dataset with known quantitative values are used and if no sharing information is available for a pair then their expected IBD sharing is taken to be 1 — this happens under 3 circumstances, (1) missing genotypes, (2) two

[11] Ref. "The investigation of linkage between a quantitative trait and a marker locus", J.K. Haseman and R.C. Elston, Behaviour Genetics 2, 3-19 (1972).

[12] This was the algorithm used by `sibdreg` in *GAS*1.4

homozygous parents, and(3) an intercross mating producing siblings who share either 0 or 2 alleles but it cannot be determined which.

If you have allele data for more than one named locus on a chromosome then, provided the recombination fraction between adjacent loci is less than 0.2, you will benefit from using the interval map version (**sibihe**) of this routine.

Example 4.3

The gasfile *qsib.gas* reads locus data from *qt_mk1.loc* and *qt_level.loc* with pedigree data from *qtrait.ped*. It calls **sibhe** and writes the results to *qsibhe.out*, then calls **sibmwu** and writes these results to *qsibmwu.out*. Plots of the points and best-fit lines for the **sibhe** regression are written to the file *qsibhe.ps*. Note the use of **fprintf** to add comments to the screen and output files.

4.9 ROUTINE : SIBTABLE

This routine displays simultaneously the sib-pair sharing across a number of affection and marker loci.

`call sibtable( locus` *locus_names options* `);`

⊙**`locus`**	list affection and marker loci to analyze
⋆ **`halfsib`**	show half-sibling data

For instance, the command

```
call sibtable( locus dis1 dis2 mk1 mk2 mk3 mk4 paternal );
```

gives a table of the allele sharing in paternally-derived chromosomes for the marker loci `mk1`, `mk2`, `mk3` and `mk4`.

4.9.1 Algorithm Notes

`sibtable` has no analytic functions, it's only purpose is to display the observed sharing in the pedigree data.

Chapter 5 : Sib-Pair Interval Mapping

Sib-pair interval mapping is a multi-point method in which information from adjacent markers is used to infer missing or ambiguous allele sharing.

5.1 Calculation of Sharing Probabilities.

The calculation of sharing probabilities is carried out in 4 steps:

[1] loci at which the sharing status is definitely known are stored,

[2] estimated sharing at ambiguous intercrosses is calculated (see below),

[3] unknown-sharing loci are interpolated (see below) from the results of [1,2],

[4] any inter-loci values are interpolated using results of [1,2,3].

Note that when the sharing at a particular locus (for a particular pair) cannot be assigned due to missing parental data, the algorithm in *GAS* calculates the expected sharing purely from the known sharing at adjacent loci rather than attempting to infer parental genotypes. This strategy was adopted to prevent incorrect results being caused by wrongly specified allele frequencies, which is a particular problem with highly polymorphic markers.

5.1.1 Ambiguous Intercrosses

With some intercrosses it can be observed that, while the actual sharing is unobservable, either (1) both paternal and maternal sharing statuses must be the same,[13] or (2) paternal and maternal sharing statuses must be opposite.[14] The expected sharing at such a locus is calculated using Bayes' formula by conditioning on the nearest adjacent loci at which sharing can be definitely assigned. To illustrate consider 3 consecutive loci X, I and Y with recombination fractions θ_{XI} and θ_{IY}. Suppose that paternal-maternal sharing is 1-0 at X, 1-1 at Y, and that I is an intercross at which the pair must either share 2 or 0 alleles IBD, then the expected paternal sharing at locus I is given by

$$\begin{aligned} P(\text{pat sharing=1 at } I \text{ |data}) &= \frac{P(\text{data}|\text{pat sharing=1 at } I)}{P(\text{data}|\text{pat sharing=1 at } I) + P(\text{data}|\text{pat sharing=0 at } I)}, \\ &= \frac{\psi_{XIm}(1-\psi_{IYm})\psi_{XIf}\psi_{IYf}}{\psi_{XIm}(1-\psi_{IYm})\psi_{XIf}\psi_{IYf} + (1-\psi_{XIm})\psi_{IYm}(1-\psi_{XIf})(1-\psi_{IYf})}, \end{aligned} \tag{5.1}$$

[13] e.g. all parents and children have genotype '1 2'.

[14] e.g. all parents and one child have genotype '1 2', other child is '1 1'.

writing $\psi_{12s} = \theta_{12s}^2 + (1-\theta_{12s})^2$, where θ_{12s} is the recombination fraction between loci 1 and 2 along the chromatid of sex s. If several intercross loci interact then Bayes' formula is extended over all the possible cases (n intercrosses generate 2^n cases) simultaneously.

5.1.2 Interpolation

Once any ambiguous intercrosses have been resolved, the paternal and maternal sharing calculations are effectively decoupled. Suppose that a, b and c are 3 adjacent loci, and that the sharing (π_a, π_c) is definitely known at the outer loci a and c, but not at b in the centre, then π_b is calculated using the formula [15]

$$\pi_b = \frac{(1-\psi_{ab})(1-\psi_{bc})(1-\psi_{ac}) - \pi_a\psi_{bc}(1-\psi_{bc})(1-2\psi_{ab}) - \pi_b\psi_{ab}(1-\psi_{ab})(1-2\psi_{bc})}{\psi_{ac}(1-\psi_{ac})}, \tag{5.2}$$

where ψ_{ij} is the sex-specific 'ψ' value between loci i and j for the chromatid pair under consideration. If b is at an end point of the region, so (for instance) a does not exist, then $\psi_{ab} = \psi_{ac} = \frac{1}{2}$ and the value of π_a is redundant. The same formula is used to interpolate intermediate values between the loci for stage [4] above.

Sib-pairs for which there is no IBD sharing information at any locus are not used by the interval mapping routines.

5.2 Parameters

5.2.1 Interval

By default the interval mapping routines infer sharing only at the actual loci listed. The **interval** parameter may be used to request that the expected sharing is calculated at points between the loci, thus adding the command option

```
call routine( ... interval 0.03 );
```

will generate extra points between the loci so that there is no region larger than $\theta = 0.03$ without such an interpolated value. Since recombination fractions cannot be added linearly (for instance twice 0.2 is 0.32) the steps taken will be smaller than the value specified after **interval**.

5.2.2 Showraw

The **showraw** parameter will display the results on a pair-by-pair basis after stage [3] of the sharing calculation described above.

[15] Based on equations in "Robust Multipoint Linkage Analysis: An Extension of the Haseman-Elston Method", J.M. Olsen, 177-193 (1995).

5.3 ROUTINE : SIBIDES

This combines interval mapping with the t_2 analysis in the **sibdes** routine.

call sibides(locus *locus_names options* **);**

⊙**locus**	list the affection and marker loci to be analyzed
⊙**theta**	list of recombination fractions between marker loci
⋆ **alltypes**	evaluate for non-affecteds also
⋆ **interval**	use interval map with specified recombination
⋆ **mapfunc**	select distance mapping function
⋆ **showraw**	display raw sharing data
⋆ **weight**	options are **strict** and **hodge**
♡	displays t_2 and $-\log_{10}(pvalue)$ along chromosome

5.3.1 Algorithm Notes

The expected sharing is calculated as described earlier in this chapter, and the 1-sided t_2 test applied to the results. Do not confuse the graph of $-\log_{10}(pvalue)$ with a lodscore statistic — the use of the logarithm is purely to allow the full range of data to be displayed on a sensible scale.

5.4 ROUTINE : SIBIHE

This routine combines interval mapping with the Elston-Haseman algorithm described in `sibhe()`.

`call sibihe( locus` *locus_names* `theta` *recom_fracs* *options* `);`

⊙**locus**	list the affection and marker loci to be analyzed
⊙**theta**	recombination fractions between markers
⋆ **absolute**	use absolute difference of values rather than square
⋆ **dfweight**	compensate for multi-pair sibships
⋆ **graph**	draw graphs of regression plots
⋆ **interval**	use interval map with specified recombination
⋆ **mapfunc**	select distance mapping function
⋆ **showraw**	display raw sharing data
⋆ **signif**	the value at which significant results are marked
♡	displays $-log_{10}(pvalue)$, also regression plots with graph option

5.4.1 Algorithm Notes

The expected sharing is calculated as described earlier in this chapter, and the Haseman-Elston test is then applied as described in routine **sibhe**. Do not confuse the graph of $-\log_{10}(pvalue)$ with a lodscore statistic — the use of the logarithm is purely to allow the full range of data to be displayed on a sensible scale.

Example 5.1

The gasfile *iplot.gas* performs the **sibides** test on the affection locus 'a1' and the **sibihe** test on the quantitative locus 'q1' using genotype data from 8 named loci labelled 1,2,...,8. The recombination values are different along the male and female chromatids. Results are written to the files *iplot.out* and *iplot.ps*.

Chapter 6 : Likelihood Calculations

The routines in this chapter are designed to perform 'traditional' linkage analysis in which alternate hypotheses about genotype/phenotype interactions are tested by computing lodscores.

All of the 'lik' routines use the Vitesse likelihood engine, which was devised and implemented by Jeff O'Connell.[16] Vitesse is the fastest likelihood program currently extant, capable of computing multipoint lodscores with highly polymorphic markers - see below for further details.

6.1 Vitesse

Jeff O'Connell's Vitesse program incorporates many new computational techniques which enable it to perform calculations impossible for other programs. In particular it is able to handle up to 8 loci simultaneously in multi-point lodscores and isn't slowed by highly polymorphic marker alleles. A more optimized (ie. faster) version of the likelihood engine is under construction and will be incorporated into *GAS* as soon as it is fully tested.

Vitesse is undergoing continuous improvement, and while we believe that all the results produced are correct, there are restrictions on the types of data it can currently handle. These are:

[1] Only a single trait locus per calculation

[2] No sex-linked loci

[3] No inbreeding or consanguinity loops

[4] Only one 'founding' nuclear unit per pedigree

Condition [4] means that there can only be one mating in any family in which all four grandparents are unknown (ie. not listed in the pedigree). Datasets which violate these conditions will cause the program to exit.

Vitesse will eventually available as a stand-alone program with a 'Linkage-like' interface[17] via anonymous ftp. Email jeff@sherlock.hgen.pitt.edu for more details on this.

---

[16] Ref. J. O'Connell et.al., forthcoming in Nature Genetics.

[17] The data and control formats are compatible with version 5.1/5.2 of LINKAGE and version 2.3P of the FASTLINK program.

6.2 Parameters

Several of the routines use a common syntax for performing particular tasks, and some of this is described below. Refer to the individual routine descriptions to see which features are available for each of them.

6.2.1 Support

Some functions are able to calculate support intervals about the location of a maximum lodscore (ie. the adjacent region where the lodscore is within a certain amount of its highest value). Hence if a lodscore has a peak of 6.3 at location X, then the support interval of height 1.5 is the adjacent region of the chromosome surrounding X on which the lodscore is continuously above 4.8

`support` *value*

If no *value* is supplied, a default of `1` is assumed.

6.2.2 Exclusion

An exclusion map shows regions of a chromosome where linkage is unlikely because the lodscore is significantly below zero. The **`exclude`** parameter is used to scan for such areas:

`exclude` *value*

If no *value* is supplied, a default of `-2` is assumed, so that any region with lodscore of -2 or lower is marked as being excluded.

6.3 ROUTINE : LIK2POINT

The `lik2point` routine performs a series of two-locus optimizations to determine the most probable recombination fractions between pairs of adjacent loci.

`call lik2point( locus` *locus_names options* `);`

⊙**locus**	list of loci to analyze
⋆ **allorders**	all possible pairs of loci are examined
⋆ **exclude**	identify exclusion region
⋆ **findmax**	find maximum lodscores in each interval between fixed loci
⋆ **mapfunc**	mapping function to use
⋆ **signif**	level to declare linkage to be significantly probable
⋆ **support**	calculate support interval
♡	plots of lodscores over range $0 < \theta \leq \frac{1}{2}$

6.3.1 Algorithm Notes

The maximization (for the **findmax** option) is carried out using Brent's algorithm, taking as starting point the highest value found during the initial scan of the range $0 < \theta \leq \frac{1}{2}$. Further parameters are available to tune the internal performance of this maximization algorithm

⋆ **initstep**

number of steps in initial scan of interval $0 < \theta \leq 0.5$, default 5.

⋆ **maxtol**

maximum tolerance in optimization, default 10^{-5}.

⋆ **maxiter**

maximum optimization iterations to attempt, default 100.

Under the vast majority of circumstances the default options will produce good results, however for 'difficult' datasets you may try increasing **initstep** and **maxiter**.

6.4 ROUTINE : LIKMAP

The `likmap` routine generates a series of likelihoods giving the probabilities that a particular locus (called 'movable') lies in various locations with respect to one or more other loci whose positions are specified (called 'fixed').

call likmap(locfix *locus_names* **locmov** *locus_names* **theta** *recombination_values options* **);**

⊙**locfix**	list of ordered fixed loci to analyze
⊙**locmov**	list of movable loci to analyze
⊙**theta**	list of recombination fractions between fixed loci
⋆ **doall**	calculate all values in subsets
⋆ **dosets**	use subsets of fixed loci of this size
⋆ **exclude**	level to indicate linkage is excluded
⋆ **findmax**	find maximum likelihood position in each interval
⋆ **mapfunc**	mapping function to use
⋆ **margin**	the minimum distance between fixed and movable loci
⋆ **showraw**	display 'actual' likelihoods
⋆ **signif**	level to declare linkage to be significantly probably
⋆ **step**	the number of steps to take between adjacent fixed loci
♡	lodscore map across the interval

6.4.1 Algorithm Notes

It is essential to set the **dosets** parameter if more than 8 fixed loci are used — otherwise the computation time and space are likely to be prohibitive. For optimal performance **dosets** should be an even number, with a value of 4 (the default) or 6 generally producing good results (also, graphical output will be messy for odd numbers of fixed loci since many points will have two values plotted). The text output displays only the two recombination fractions to either side of the movable locus, since for each order the others are fixed by the input parameters.

The maximization is carried out using Brent's algorithm, taking as starting point the highest value found whilst constructing the map (the resolution of which may be changed using the **step** parameter).

Example 6.1

The gasfile *twop.gas* uses `lik2point` to calculate the most-probable recombination fractions between a series `mka–d` of 4 adjacent marker loci. Distances are given in Morgans (M) using the Kosambi map.

Example 6.2

The gasfile *map.gas* demonstrates the use of `likmap` to create a table showing the likelihood of the loci `try1` and `try2` being at various locations along a chromosome on which the five markers `mka–e` have previously been positioned. Distances are shown using the Haldane map.

6.5 ROUTINE : LIKSINGLE

The **liksingle** routine performs a single likelihood calculation for a fixed set of loci and recombination fractions.

call liksingle(locus *locus_names* **theta** *recombinations* *options*);

⊙**locus**	list of ordered loci to analyze
⊙**theta**	list of recombination fractions between loci
⋆ **genlod**	computes Ott's generalized lodscore

6.5.1 Algorithm Notes

Because of internal variations in algorithms, the likelihoods calculated by two programs for the same dataset may vary enormously. However the ratio of two likelihoods (as computed by the same program) should be invariant between programs, and thus the lodscores produced by such programs should be very similar.

The generalized lodscore compares the likelihood against the value when all the recombination fractions are set to $\frac{1}{2}$. For a single pair of loci it is identical to the normal lodscore.

Example 6.3

The gasfile *sin.gas* demonstrates the use of **liksingle** to show the likelihood of **mka**, **mkb** and **mkc** lying on the same chromatid separated by recombination fractions $\theta = 0.35$ and 0.31.

Chapter 7 : Association Analysis

The routines for association analysis look for correspondences between the occurrences of particular alleles of named loci and the values of traits in the population.

To perform association tests it is essential that the names of alleles be the same in different families (eg. named allele '1' must represent the same physical marker in the whole population). This means that global binning (preferably with fixed bin sizes) must be used if data is read using the **`alsize`** option.

7.1 ROUTINE : ASSTDT

The **asstdt** routine performs association analysis between a marker and an affection locus using the Transmission Dis-equilibrium Test.[18]

`call asstdt( locus` *locus_names options* `);`

⊙**locus**	list affection and marker loci to analyze
⋆ **sexual**	show separate analysis for paternal and maternal alleles
⋆ **signif**	set significance criteria
⋆ **weight**	reduce contribution of multiple sibships

7.1.1 Algorithm Notes

The standard algorithm follows Spielman's advice of treating all children as independent observations, summing their transmitted and non-transmitted alleles, and calculating the significance using the exact 1-sided binomial distribution.

Some authors suggest that only one child should be used from each mating, and that this child be selected according to fixed ascertainment criteria. To employ this strategy you need to remove the other children from the pedigree file before running **asstdt**. If there are several children within a family which satisfy the ascertainment criteria equally well (so that selecting a particular one would be arbitrary), then the **weight** option will calculate the average contribution from each of these 'equivalent' children and treat this as being the contribution due to a single child. Since the **weight** option may result in non-integer totals, the χ^2 distribution (with 1 degree of freedom) is used to calculate the significance.

[18] Ref. "Transmission Test for Linkage Disequilibrium: The Insulin Gene Region and Insulin-Dependent Diabetes Mellitus", R.S. Spielman, R.E. McGinnis, W.J. Ewens, Am.Jou.Hum.Gen. 52:506:516 (1993).

7.2 ROUTINE : ASSCOMPARE

The **asscompare** routine compares the allele frequencies between two groups of subjects denoted by `y`/`n` values at an affection locus.

`call asscompare( locus` *locus_names options* `);`

⊙**locus**	list of affection and marker loci to analyze
⋆ **sexual**	show males and females separately
⋆ **signif**	set significance criteria
⋆ **useall**	use half-known genotypes

7.2.1 Algorithm Notes

For a locus with n alleles, *GAS* constructs a $2{\times}n$ contingency table showing how often each allele occurs in the two populations (ie. the sets of people labelled `y` and `n`). A χ^2 test is used to assess to what extent the allele frequencies differ between the populations. Each of the alleles is also tested individually by grouping all of the other alleles into a single bin and performing χ^2 tests on the $2{\times}2$ contingency tables produced.

7.3 ROUTINE : ASSMWU

The `assmwu` routine performs association analysis between a marker and a quantitative locus, using the Mann-Witney U-Test. [19]

`call assmwu( locus` *locus_names options* `);`

⊙`locus`	list of quantitative and marker loci to analyze
⋆ `allinfo`	give extra information
⋆ `cutoff`	cutoff for displaying p-values
⋆ `exact`	sizes of dataset below which p-values calculated exactly
⋆ `sexual`	show separate results by subject sex
⋆ `signif`	set significance criteria

7.3.1 Algorithm Notes

Two tests are performed. The first treats each allele as a separate observation, so that a subject with genotype '`1 3`' appears in both the '`1`' and '`3`' allele categories, and a homozygous subject appears twice in the same category. Subjects with half-known genotypes (eg. '`1 x`') contribute a single observation. The alleles are ranked one at a time (ie. each allele versus all the others) according to the quantitative values associated with them.

The second test categorizes subjects according to whether they do or do not have a particular allele. The ranks of the subjects (according to the quantitative trait) who have each allele are compared with those who do not have the allele to indicate if the allele tends to be associated with subjects who are biased in a particular direction away from the mean. Subjects with half-known genotypes are not used.

Note that exact calculation of p-values requires a large amount of time and memory (RAM is approximately proportional to $N^2M^2/4$ where N, M are the datasets being compared) and the optimal values for `exact` will depend on your computer. If *GAS* halts with an out-of-memory message, reduce one or both of the `exact` values. For example, the command

```
call assmwu( locus weight mar1 exact 20 50 );
```

[19] The Wilcoxon Rank-Sum test is equivalent to the Mann-Witney U-test.

performs the Mann-Witney U-test on the quantitative locus `weight` against the marker `mar1`. P-values are calculated by a Gaussian approximation unless there are less than 20 instances of a particular allele versus a set of 50 instances of other alleles.

Example 7.1

The gasfile *assoc.gas* reads pedigree data from the file *assoc.ped*. The `asstdt` routine is used on the loci `disease` and `marker1`, and the `assmwu` test is used on `response` and `marker1`. The results are written to the files *tdt.out* and *mwu.out* respectively.

7.4 ROUTINE : ASSRELPREF

The **assrelpref** routine performs association analysis between a marker and an affection locus, using the Relative Predispositional Effect technique.[20]

`call assrelpref( locus` *locus_names options* `);`

⊙**locus**	list of quantitative and marker loci to analyze
⋆ **alltypes**	results are shown for non-affected subjects
⋆ **signif**	set significance criteria
⋆ **sexual**	males and females are analyzed separately

For example, the command

```
call assrelpref( locus spotty mar1 );
```

performs the RPE analysis on the affection locus `spotty` in terms of the alleles of the marker locus `mar1`.

7.4.1 Algorithm Notes

The RPE method calculates a p-value for each allele individually according to the formula $\chi^2 = (O_i - E_i)^2/E_i$, where O_i is the observed number of occurrences of allele i in the dataset, and E_i is the number expected according to the input allele frequencies. The total p-value for the dataset is calculated by adding together the non-zero alleles and using a χ^2 test with degrees of freedom one less than the number of non-zero alleles.

If the total p-value is less than the significance criteria (which may be altered with the **signif** parameter) then the allele with the smallest p-value is removed from the dataset and the expected frequencies are re-calculated as though that allele did not exist. This procedure is then repeated until the total p-value becomes non-significant.

[20] Ref. "RPEs of Marker Alleles with Disease: HLA-DR Alleles and Graves Disease" Payami et.al., Am.J.Hum.Genet 541-546 (1989).

7.5 ROUTINE : ASSGENORR

The **assgenorr** routine performs association analysis between a marker and a condition using the Genotype Relative Risk method,[21] in which the observed distribution of named alleles in a subset of the pedigree is compared to that predicted from the input allele frequencies (entered earlier using the **set locus** command) under the assumption of Hardy-Weinberg equilibrium.

call assgenorr(locus *locus_names* **allele** *allele_names options* **);**

⊙**locus**	list of marker (and optionally affection) loci to analyze
⊙**allele**	list of alleles of the marker loci to analyze
⋆ **inpairs**	compare two single genotypes
⋆ **incommon**	compare genotype against others with allele in common
⋆ **allother**	compare genotype against all others
⋆ **signif**	set significance criteria

If no affection loci are listed, then the whole of the pedigree is compared to the input allele frequencies, and the risks computed refer to the probability of a random member of the population being selected to form part of the dataset (for optimum performance the members of the pedigree should not be related).

The parameters **inpairs**, **incommon**, and **allother** may be combined in a single command.

7.5.1 Inpairs

The **inpairs** option tests listed pairs of alleles against each other. For example, the command

```
call assgenorr( locus spotty mar1 allele 1 2 3 4 inpairs );
```

calculates the relative risk of subjects having genotype '1 2' at locus mar1 of being affected at locus spotty, compared to subjects with genotype '3 4'. More than two pairs of alleles can be listed.

[21] Ref. "Estimating Genotype Relative Risks", M. Lathrop, Tissue Antigens 22, 160-166 (1983).

7.5.2 Incommon

The `incommon` option tests specific allele pairs against all the haplotypes sharing a particular allele in common with them. The command

```
call assgenorr( locus mar2 allele a1 a2 incommon );
```

calculates (separately) the relative risks of subjects having genotype 'a1 a1', 'a1 a2' and 'a2 a2' at locus mar2 of being affected at locus spotty, compared to subjects sharing one of these alleles.

7.5.3 Allother

The `allother` option tests specific allele pairs against all other allele pairs simultaneously. The command

```
call assgenorr( locus hairy mar3 allele alpha beta allother );
```

calculates (separately) the relative risks of subjects having genotype 'alpha alpha', 'alpha beta' and 'beta beta' at locus mar3 of being affected at locus hairy, compared to subjects having any other combination of alleles.

7.5.4 Algorithm Notes

The genotype relative risk (R) of genotype set A individuals compared to genotype set B individuals is calculated according to the formula

$$R_{AB} = \frac{\left(O_A + \frac{1}{2}\right) F_B}{\left(O_B + \frac{1}{2}\right) F_A},$$

where O_A and O_B are the respective observed counts of the genotypes in the dataset, and F_A and F_B are the expected counts based on the input allele frequencies under the assumption of Hardy-Weinberg equilibrium.

Example 7.2

The gasfile *grr.gas* reads pedigree data from *assoc.ped* and uses the **assgenorr** method to compare the haplotype 2 3 (at locus 'marker') against 1 3 (**inpairs** results in *grrpair.out*), to compare the haplotype 1 2 against all other haplotypes having allele 1 then against all other haplotypes having allele 2 (**incommon** results in *grrcom.out*), and lastly to compare haplotype 3 3 against every other possibility for the subjects affected at locus 'disease' (**allother** results in *grrall.out*).

7.6 ROUTINE : ASSEMPLOG

The **assemplog** routine performs association analysis between a pair of named or affection loci and a condition using the Empirical Logistic method [22]

call assemplog(locus *locus_names options* **);**

⊙**locus**	list of marker and affection loci to analyze
⋆ **showraw**	display raw Z terms and variances
⋆ **signif**	set significance criteria

At least three loci (one of which must be an affection status) must be listed. If named loci are listed then all alleles are tested in a pairwise fashion.

7.6.1 Algorithm Notes

In cases where there are no subjects possessing a particular genotype/phenotype combination, the variance of the statistic becomes infinite and a p-value of 1 is returned.

Example 7.3

The file *elm.gas* read data from *elm.loc* and *elm.ped* and performs the **assemplog** analysis. (the subjects are all 'singletons' and *GAS* will generate warnings about this — press 'c' to continue at each stage)[23] Results are writen to the files *elm.1a*, *elm.1b* and *elm.1c*.

[22] Ref. "Association between HLA Antigens and the Presence of Certain Diseases", J.R. Green et.al., Statistics in Medicine Vol.2, 79-85 (1983).
[23] If you do **set maxwarnings** to a large value, most of the repeated questioning will cease.

Chapter 8 : Haplotyping

Haplotyping is the process of determining which alleles in an un-ordered genotype are descended from each of a subjects parents, and thus (when this is done for several linked loci) re-constructing segments of the chromatids within each subject.

8.1 ROUTINE : HAPCHILD

The **hapchild** routine determines the allelic phase of the genotypes of children within the input population. For those children at which the phase can definitely be decided for all or their alleles, the observed haplotypes are ordered in decreasing frequency. The syntax is:

call hapchild(locus *locus_names options* **);**

⊙**locus**	list affection and marker loci to analyze
⋆ **sexual**	show separate analysis for paternal and maternal chromatids

8.1.1 Algorithm Notes

The **hapchild** routine only uses the alleles for which the parental origin can be definitely determined — there is no attempt to assign probabilities to ambiguous cases (which are marked '**x**' and ignored when counting haplotype frequencies). The haplotypes are listed in order of decreasing frequency.

Example 8.1

The gas-file *chap.gas* loads pedigree data from *chap.ped* and uses **hapchild** to calculate the most frequently occurring haplotypes in the children. Results are sent to the file *chap.out*.

End of Gas Manual version 2.0