

GENETIC MAPS & QTL ANALYSIS

Part 1 - GENETIC MAP

The goals of this exercise are to:

1. Understand one software package available for the creation of genetic maps
2. Use real data sets to find linkage groups by two-point linkage
3. Explore genetic map orders by hand
4. Display a genetic map

Source code, updates, and other information are available on the Internet from:
<http://www.broadinstitute.org/ftp/distribution/software/mapmaker3/>

Software Overview:

MAPMAKER/EXP is a linkage analysis package designed to help construct primary linkage maps of markers segregating in experimental crosses. MAPMAKER/EXP performs full multipoint linkage analysis (simultaneous estimation of all recombination fractions from the primary data) for dominant, recessive, and co-dominant markers.

1. Starting MAPMAKER/EXP 3.0

Connect to plantgenome.plantsciences.ucdavis.edu using **Putty**. You will not need to run Xming OR enable X11 forwarding since this program will not involve the use of graphical display.

Once connected, use WinSCP to transfer the Lab9 folder onto the server.

Now enter the directory for part 1 of the lab (P1);

```
>cd Lab9/P1
```

This directory should contain one file with the extension .raw. This file is a specific format required by **MapMaker**. The content in this file will be required for the analysis.

```
>ls
```

Raw files are flat ASCII text files which may be generated in many ways but must use a straight text editor and NOT a word processor such as MS Word. Notepad (Windows), Jedit (all platforms), or BBedit (Mac) are all acceptable programs.

MAPMAKER tends to be very lenient about how you separate items in a data file (e.g. spaces, tabs, or sometimes line breaks), and is generally insensitive to extra spaces, uppercase-lowercase distinctions, and (after the top two lines) blank lines.

It is still possible to format a file in such a way as it confuses MAPMAKER. If you have trouble, try to make your MAPMAKER file look more like the sample.raw file in the Lab7 directory.

The very first line of your raw data file should read like:

data type xxxx

where xxxx is one of the allowed data types:

f2 intercross
f2 backcross
f3 self
ri self
ri sib

The second line of the raw file should contain a list of three numbers, each separated by a single space:

46 362 2

The first of these values indicates the number of progeny for which data are included in the file (in this case, 46). The second indicates the number of genetic loci for which data are supplied (362). The third indicates the number of quantitative traits in the data set (here 2, although this may be zero)

Additional information may be optionally supplied at the end of this line. In particular, you may specify the coding scheme you use for genotypes.

By default, the codes used for F2 backcross data are:

'A' Homozygote for the recurrent parent genotype.
'H' Heterozygote.
'-' Missing data for the individual at this locus.

For F2 intercross data, the default codes are:

'A' Homozygote for the allele from parental strain a of this locus.
'B' Homozygote for the allele from parental strain b of this locus.
'H' Heterozygote carrying both alleles a and b.
'C' Not a homozygote for allele a (either bb or ab genotype.)
'D' Not a homozygote for allele b (either aa or ab genotype.)
'-' Missing data for the individual at this locus

For RI data, the default codes are:

'A' Homozygote for parental genotype a.
'B' Homozygote for parental genotype b.
'-' Missing data for the individual (or line) at this locus.

Also by default, MAPMAKER will match genotype characters in a case-insensitive manner (that is 'a' and 'A' indicate the same genotypes). However, you can tell MAPMAKER to use whatever conventions you like, so long as you use the same conventions for the entire data file.

If you follow the numbers on the second line with the word "case", then MAPMAKER will match genotype characters in a case sensitive manner (that is 'a' and 'A' can be used to indicate different genotypes). For example:

```
46 362 2 case
```

To specify the coding scheme itself, include on the end of the above line the word "symbols" followed by the coding scheme you wish to use, defined in terms of the coding scheme above.

For example, if you wish to use the following scheme with an RI data set:

'1' Homozygote for parental genotype
'2' Homozygote for parental genotype
'0' Missing data for the individual (or line) at this locus

Use a second line:

```
46 362 2 symbols 1=A 2=B 0=-
```

The main restriction on coding schemes are that the permitted symbols: letters, numbers, and the characters '-' and '+'.
'

After the first two header lines, the raw file should then present the genetic locus data, in the following simple format:

For each locus, list:

- (1) the name of the locus, preceded by an asterisk ("**")
- (2) one or more spaces (or tabs etc.)

(3) the genotypic data for all individuals, in order.

For example:

```
*locus1 BA-HHHAAABBB-HHAA
```

individual #1 having the B genotype

individual #2 having the A genotype

...

Data for each new locus should begin on a new line (with blank lines allowed),

Locus names should be kept to at most 8 characters, and must be limited to alphabetic and numeric characters, along with the underscore character ('_') and periods ('.'). Locus names must start with an alphabetic character

Any quantitative trait data should come after the genetic locus data. These data follow a similar format, except that the trait values for each individual must be separated by at least one space, tab, or line break.

A dash ('-') indicates missing data.

```
*weight 6.3 7.7 8.0 6.2 8.6 - 7.5 9.0 5.5 - - 8.4 7.7 7.4 6.9 -
```

Corresponds to a trait named "weight", for which individual #1 has a value of 6.3, individual #2 has a value of 7.7, and so on. The sixth individual is missing data for this trait (and will be ignored for all analyses involving these trait data).

As for the genotypes, a new trait should begin on a new line, and line breaks are allowed. Traits may also be specified as functions of other existing trait data.

```
*weight1 6.3 7.7 8.0 6.2 8.6 6.9 7.5 9.0  
*weight2 6.7 7.9 7.5 6.8 8.0 7.3 7.5 9.5  
*mean= (weight1 + weight2)/2
```

The format of these equations is described under the "make trait" command. Such traits must be included in the number of traits indicated on the file's second line.

Take a look at the .raw file that has been created for you:

>[more SAMPLE.RAW](#)

start **MAPMAKER**:

>mapmaker

When **MAPMAKER** starts, you will first see its start-up banner and a prompt **1>** for the first command. This prompt is not the standard UNIX prompt but operates within the software program:

2. Prepare Data as Input to MapMaker

The first step in almost every **MAPMAKER** session is to **prepare and load a data file for the analysis**. If you are starting out an analysis on a new data set, or if you have modified the raw data in an existing data set, you will do this using **MAPMAKER's prepare data** command.

If instead you are resuming an analysis of a particular (unmodified) data set, you may use the **load data** command, which preserves many of the results from your previous session.

Since we are just starting, we will use **MAPMAKER's prepare data** command to load our sample data file `sample.raw` located in the **Lab7** directory:

>prepare data sample.raw

```
1> prepare data sample.raw
preparing data from file 'sample.raw'... ok
  F2 intercross data (333 individuals, 12 loci)... ok
unable to run file 'sample.prep'... skipping initialization
saving genotype data in file 'sample.data'... ok
saving map data in file 'sample.maps'... ok
saving traits data in file 'sample.traits'... ok
2> █
```

From this file, **MAPMAKER** extracts the type of cross, the number of scored progeny, the number of used markers, and the genotype for each marker in each individual (if available). Other information may be present in the data files, such as quantitative trait data and pre-computed linkage results.

Before performing any analysis, we now instruct **MAPMAKER** to **save a transcript of this session in a text file** for later reference. Using the **photo** command, we start a transcript named `tutorial.out`.

Note, if the file already exists, **MAPMAKER** appends a new output to this file.

> [photo tutorial.out](#)

```
2> photo tutorial.out
'photo' is on: file is 'tutorial.out'
3> █
```

3. Finding Linkage Groups by Two-Point Linkage

Find linkage groups by performing a classical **two-point, or pairwise, linkage analysis** on our data file. To do this we need to **tell MAPMAKER which loci we wish to consider in our two-point analysis**, which we do by using **MAPMAKER's sequence** command.

MAPMAKER is told which loci (and, in some cases, which order for those loci) any following analysis command should consider.

Since almost all of **MAPMAKER's** analysis commands use the current sequence to indicate which loci they should consider, you will find that **the sequence command must be entered before performing analysis**.

The sequence of loci remains final until the **sequence** command is run again to modify it.

In this two-point analysis we want **MAPMAKER** to examine all 12 of the loci present our sample data file:

> [sequence 1 2 3 4 5 6 7 8 9 10 11 12](#)

```
3> sequence 1 2 3 4 5 6 7 8 9 10 11 12
sequence #1= 1 2 3 4 5 6 7 8 9 10 11 12
4> █
```

Note that **for two-point analysis, the order in which the loci are listed is not important**.

MAPMAKER's group command, instructs the program to **divide the loci into linkage groups**.

To determine whether any two markers (loci) are linked, **MAPMAKER** calculates the maximum-likelihood distance and corresponding LOD score between the two markers. If the LOD score is greater than some threshold, and if the distance is less than some other threshold, then the two markers will be considered linked.

By default, the LOD threshold is 3.0, and the distance threshold is 80 Haldane cM.

For the purpose of finding linkage groups, **MAPMAKER** considers linkage transitive. That is, if marker A is linked to marker B, and if B is linked to C, then A, B, and C will be included in the same linkage group.

Type the group command:

```
> group
```

```
4> group
Linkage Groups at min LOD 3.00, max Distance 50.0

group1= 1 2 3 5 7
-----
group2= 4 6 8 9 10 11 12

5> █
```

As you see, **MAPMAKER** has divided the 12 loci into 2 linkage groups, named as **group1** and **group2**.

There is no unlinked marker in this data file.

4. Exploring Map Orders by Hand

To **determine the most likely order of the markers within a linkage group**, we could imagine using the following simple procedure:

For each possible order of markers within that group, we calculate the maximum-likelihood map (e.g. the distances between all markers), and the corresponding likelihood of the map. We compare these likelihoods and choose the most likely order as the answer. This type of exhaustive analysis may be performed using **MAPMAKER's compare** command.

In practice, this is not practical for even medium sized groups: a group of N markers has $N!/2$ possible orders, a number which becomes unwieldy when N gets to be more than 10.

In practice, one needs to order subsets of the linkage group and then overlap those subsets, mapping any remaining markers relative to those already mapped.

Because the linkage groups (group 1 and group2) are small, we can use a fully **exhaustive analysis**.

To do an exhaustive analysis for marker order starting with linkage group1 , we first need to **change MAPMAKER's sequence** to the one of group1:

➤ `sequence {1 2 3 5 7}`

*remember curly brackets

```
5> sequence 1 2 3 5 7
sequence #2= 1 2 3 5 7

6> █
```

Again, **the order of markers here does not matter**. Here, the **curly brackets indicate that the order of the markers contained within them is unknown**, and thus all possible orders between markers need to be considered.

We then type the **compare** command, instructing **MAPMAKER** to compute the maximum-likelihood map for each possible order of markers, and to report the orders sorted by the likelihoods of their maps.

> `compare`

```
6> compare

Best 20 orders:
1:   1 3 2 5 7   Like:  0.00
2:   3 1 2 5 7   Like: -6.00
3:   5 7 2 3 1   Like: -20.20
4:   5 7 2 1 3   Like: -26.26
5:   2 5 7 3 1   Like: -27.25
6:   2 5 7 1 3   Like: -28.39
7:   2 3 1 5 7   Like: -28.85
8:   5 2 3 1 7   Like: -32.33
9:   2 1 3 5 7   Like: -34.12
10:  5 7 1 3 2   Like: -35.55
11:  5 2 1 3 7   Like: -37.61
12:  1 3 5 2 7   Like: -37.76
13:  3 1 5 2 7   Like: -39.09
14:  5 7 3 1 2   Like: -40.38
15:  1 3 5 7 2   Like: -40.87
16:  3 1 5 7 2   Like: -41.55
17:  5 2 7 3 1   Like: -43.67
18:  5 2 7 1 3   Like: -44.78
19:  5 1 3 2 7   Like: -47.63
20:  2 5 3 1 7   Like: -52.28

order1 is set

7> █
```

The smaller the log-likelihood of a map, the more probable that the order of markers is the one provided by that map. The best first order of markers: **1 3 2 5 7** is indicated as having a relative log-likelihood of 0.00.

The second order of markers: 3 1 2 5 7 is significantly less likely than the first one, having a relative log-likelihood of -6.0. Said in a different way, the best first order of markers for this linkage group1 is supported by an odds ratio of 1,000,000:1 (10 to the 6th power to one), over any other order.

We consider this as good evidence that we have found the right order.

5. Displaying a Genetic Map

When we used the **compare** command previously, **MAPMAKER** calculated the map distances and log-likelihoods for each of the 60 possible orders of markers (5 markers = $(5*4*3*2*1)/2 = 60$).

The compare command only reports the relative log-likelihoods, and afterwards forgets the map distances. **To actually display the genetic distances** we must use the **map** command.

The map command instructs **MAPMAKER** to calculate the maximum-likelihood map of each order specified in a sequence. If the sequence specifies more than one order (for example, the sequence {1 2 3 5 7} specifies 60 different orders) then the maximum-likelihood maps for all specified orders will be calculated and displayed.

Because we found one order of this group to be much more likely than any other, we probably only care to see the map distances for this single order.

First, we set **MAPMAKER**'s sequence, putting the markers of linkage group1 in their best order (no curly brackets):

```
> sequence 1 3 2 5 7
```

```
7> sequence 1 3 2 5 7
sequence #3= 1 3 2 5 7
8> █
```

Next, we simply use the **map** command to display this order's maximum likelihood map:

```
> map
```

```
8> map
=====
Map:
Markers          Distance
 1 T175           4.2 cM
 3 C35            15.0 cM
 2 T93            11.9 cM
 5 C66            12.2 cM
 7 T50B          -----
                   43.2 cM   5 markers   log-likelihood= -424.94
=====
```

As you can see, the distances between neighboring markers are displayed.

These distances may be considerably different than the two-point distances between those markers because **MAPMAKER**'s so-called multipoint analysis facility can take into account much more information, such as flanking marker genotypes and some of the missing data.

This is the reason that we use multipoint analysis rather than two point analysis to order markers: *more data is taken into account, you have a smaller chance of making a mistake*

6. More Information

tmap: <http://math.berkeley.edu/~dustin/tmap/>

CarthaGene: <http://www.inra.fr/mia/T/CarthaGene/>

JoinMap: <http://www.kyazma.nl/index.php/mc.JoinMap>

Part 2 - QTL ANALYSIS

Quantitative Trait Loci:

Often traits in plants and animals are influenced by many genes rather than a single locus. These traits are termed quantitative traits and the loci that control these traits - quantitative trait loci, abbreviated as QTLs. An important goal in genetics and breeding is to identify and characterize QTLs, especially those that contribute to variation in quantitative traits both within and between populations or species.

Genetic linkage maps based on molecular markers can span the genome at regular intervals. The experimenter can then look for correlations between these mapped markers and the trait of interest in controlled breeding experiments to gain insight into the regions of the genome that control the trait.

Software:

QTL Cartographer is a collection of programs intended for mapping the location of Quantitative Trait Loci (QTL) in inbred populations using a map of molecular markers. To create the inbred population, you need to begin with two completely inbred parental lines, which are crossed to produce an F1 population. From here you have a great deal of freedom. This package can handle backcrossing, selfing, double haploid crosses, test crosses, and more.

<http://statgen.ncsu.edu/qtlcart/manual/>

QTL Cartographer contains five main programs:

- **QStats:** Quantitative statistics
- **LRmapqtl:** Linear regression
- **SRmapqtl:** Stepwise regression
- **Zmapqtl:** Interval mapping
- **JZmapqtl:** Interval mapping for multiple traits

QStats is a simple program to compute the sample size, mean, variance, standard deviation, skewness, kurtosis, and average deviation from your data set. It also includes a simple histogram of the trait values for visual analysis.

LRmapqtl uses a simple **linear regression model to map quantitative trait loci to a map of molecular markers**. For each marker, it fits the phenotypic data to the linear model:

$$y_i = b_0 + b_i x_i + e,$$

where y_i is the phenotype of the i^{th} individual, and x_i is the marker genotype. The regression parameters b_0 and b_i can be estimated, and e is the error assumed to have a normal distribution. The program produces the computed parameters for each linear model for each marker, as well as the corresponding p -value (using an F statistic).

SRmapqtl uses the technique of **stepwise regression to search for QTLs**. This process ranks all markers according to their effect on the quantitative trait. The analysis can be performed forwards (adding markers to the model) or backwards (deleting markers from the model). In either case, an F statistic is used to determine relevance.

Zmapqtl performs **either interval or composite interval mapping**. A likelihood ratio test statistic is generated for each region tested. A number of different models can be tested using this program. The most standard approach is not to use any markers to control for genetic background (model 3). For designs capable of estimating additive and dominance effects, there are four hypotheses: both the additive and dominance effects are either equal to, or not equal to, 0.

JZmapqtl expands the previous program to **multiple traits analyzed simultaneously**. In addition, certain instances of gene x environment interactions can be built into the models tested by this program.

1. Example of QTL mapping using QTL Cartographer

Using **Putty**, login to **plantgenome.plantsciences.ucdavis.edu**. You will need to run **Xming** in advance and forward X11 on **Putty** to view the graphical output.

Move to the sub-directory for part 2 (P2):

In this directory, there will be a folder called **qwork**. Navigate into the **qwork** folder and list the contents:

```
>cd qwork
```

```
>ls
```

The following folders will be present:

laurie

mackay

mletest

rauh

realdat

sample

sim

weber

Navigate into the **sample** folder and list the content:

```
>cd sample
```

```
>ls
```

You should find the following two files:

sample2.raw (contains the **genotypic (marker) and phenotypic (trait) information** for each individual of the inbred population)

sample.mps (contains information for the **genetic map**)

Enter the following commands:

```
>Rmap -i sample.mps -X sample
```

Enter the number 0 when prompted for a value (use this for all runs)

Rmap can either simulate a genetic map ('random **map**') or translate genetic map information from different formats into that required by QTL Cartographer ('reformat **map**').

To simulate a map, you can specify the number of chromosomes, markers per chromosome, and average inter-marker distance for the simulation. This would yield a simulated map that better approximates one that you might actually produce.

By default, the mapping function used to simulate the genetic map from the provided information is Haldane. **-i** is the command specified for the input file

'sample.mps', and **-x** is the command specified for the output file, which will be called '**sample**' and have the extension '**.map**'.

```
=====
QTL Cartographer v. 1.17j, 28 January 2005 for Unix
Copyright (C) 1996-2001 C. J. Basten, B. S. Weir and Z.-B. Zeng.
QTL Cartographer comes with ABSOLUTELY NO WARRANTY.
This is free software, and you are welcome to redistribute it
under certain conditions. For details see the file COPYING.
=====
No.          Options                               Values:
-----
0. Continue with these parameters
1. Input File                sample.mps
2. Output File               sample.map
3. Error File                sample.log
4. Random Number Seed       1192901655
5. Map Function [1,8], 1 => Haldane      1
6. Map Function Parameter    0.000000
7. Ouput (1,2,3) => Text, Graphics, Both 1
-----
8. Specify Resource File     qtlcart.rc
9. Change Filename stem     sample
10. Change Working Directory:
11. Quit
12. Quit, but update the Resource File
=====
Please enter a number... █
```

Enter the following commands:

>[Rcross -i sample2.raw](#)

Rcross uses the information generated by **Rmap** and randomly simulates a data set.

Compute the statistics from your data:

>[Qstats](#)

Qstats will generate a file named '**qtlcart.qst**'. View the summary of the statistics computed from your data:

>[more sample.qst](#)

This is for -trait 1 called weight

Sample Size.....	304
M(1).....	6.1055
M(2).....	49.5741
M(3).....	521.2409
M(4).....	6798.3551
Mean Trait Value.....	6.1055
Variance.....	12.3379
Standard Deviation.....	3.5125
Coefficient of Variation...	0.5753
Average Deviation.....	2.5944
Skw..LW(24).....	69.0864
.....Sqrt(6/n).....	0.1405
Kur..LW(29).....	1010.8460
.....Sqrt(24/n).....	0.2810
k3..LW(24).....	1.5942
k4..LW(28).....	3.6406
S (5%: 5.99, 1%: 9.21).....	296.6418

This is for -trait 2 called lnweight

Sample Size.....	305
M(1).....	1.6559
M(2).....	3.0495
M(3).....	6.0484
M(4).....	12.7331
Mean Trait Value.....	1.6559
Variance.....	0.3086
Standard Deviation.....	0.5555
Coefficient of Variation...	0.3355
Average Deviation.....	0.4361
Skw..LW(24).....	-0.0202
.....Sqrt(6/n).....	0.1403
Kur..LW(29).....	0.2934
.....Sqrt(24/n).....	0.2805
k3..LW(24).....	-0.1176
k4..LW(28).....	0.0817
S (5%: 5.99, 1%: 9.21).....	0.7876

- What is the first trait?
- Record its mean, variance, standard deviation, and coefficient of variation.
- Look at the histogram. Is the trait normally distributed?
- What is the second trait?
- Look at the histogram. Is the trait normally distributed?

Perform a linear regression analysis for each individual marker to test whether a marker is linked to a QTL:

>LRmapqtl

LTmapqtl will generate a file named 'sample.lr'. To look at the estimated parameters of the linear regression model and the F values, view this file:

>more sample.lr

```

This output is based on the map in (sample.map)
And the data in (sample.cro)

Sample Size..... 333

This analysis fits the data to the simple linear regression model
y = b0 + b1 x + e
The results below give the estimates for b0, b1 and the F statistic
for each marker.

We are interested in whether the marker is linked to a QTL. We test
this idea by determining if b1 is significantly different from zero. The F
statistic compares the hypothesis H0: b1 = 0 to an alternative H1: b1 not 0.
The pr(F) is a measure of how much support there is for H0. A smaller pr(F)
indicates less support for H0 and thus more support for H1. Significance at
the 5%, 1%, 0.1% and 0.01% levels are indicated by *, **, *** and
****, respectively.

Note that our Likelihood ratio test statistic compares two nested hypotheses
and is two times the negative natural log of the ratio of the likelihoods. For
example,
assume that hypothesis H0 is nested within H1 and that they have likelihoods L0
and L1 respectively.
Then, the "Likelihood Ratio Test Statistic" is -2ln(L0/L1).
# This trait is: weight, and
-t 1 is the number of trait being analyzed.
-----

```

Chrom.	Marker	b0	b1	-2ln(L0/L1)	F(1,n-2)	pr(F)
1	1	5.546	1.318	12.177	12.327	0.001 ***
1	2	5.590	1.228	11.019	11.136	0.001 ***
1	3	5.649	1.431	20.330	20.837	0.000 ****
1	4	5.780	1.412	20.523	21.041	0.000 ****
1	5	5.784	1.191	13.020	13.198	0.000 ***
2	1	6.084	1.443	23.575	24.283	0.000 ****
2	2	6.012	1.496	26.485	27.402	0.000 ****
2	3	6.113	1.729	33.470	34.999	0.000 ****
2	4	6.070	1.462	25.512	26.356	0.000 ****
2	5	6.087	0.562	3.799	3.797	0.052
2	6	6.104	0.749	5.133	5.141	0.024 *
2	7	6.058	0.214	0.568	0.565	0.453

What analysis has been performed to test whether a marker is linked to a QTL?

- Which chromosomes show markers significantly linked to a QTL?
- Which is the significance level of the markers that show linkage to a QTL?

Perform a stepwise regression analysis:

>[SRmapqtl](#)

SRmapqtl will generate a file named '**sample.sr**'. View the output:

>[more sample.sr](#)

```
-FB Stepwise regression analysis for -trait 1
# 2 DOF in the numerator: The given DOF is for the denominator...
-----
Chromosome   Marker   Rank   F-Stat   DOF
-----
           1         4       2     34.91059   299
           2         3       1     38.13607   301
-----
                                -start
                                -end
-----
```

The first two columns indicate the chromosome and marker. The third column gives the rank of that marker as determined by the stepwise regression analysis. The *F* statistic indicates the difference between having that variable in the model or not. Finally, the DOF (degrees of freedom) for the numerator of that *F* statistic is given.

- Which are the markers that seem to be more closely linked to a QTL?

Perform an interval mapping:

>[Zmapqtl -M 3](#)

Zmapqtl will generate a file named '**sample.z**'. **-M** is the command specified for the model, and the default model **3** does not use any markers to control for the genetic background. This is also known as **interval mapping**, and is the same as Lander and Botstein's method. To look at the interval mapping results, enter the following command:

>more sample.z

Graphically display your results.

>Eqtl -S 12.0

>Preplot

>gnuplot sample.plt

Eqtl is a utility that quickly picks out the possible QTLs from the results of **Zmapqtl**. **-S** is the command specified for the threshold (12.0) that will be used to estimate the possible QTLs.

Preplot reformats the output of the analysis programs so that it can be plotted by **Gnuplot**.

Gnuplot is free plotting software available for UNIX, Macintosh, and Windows machines.

- Look at the plots. Which are the markers that seem to be linked to a QTL?
- Does this answer agree with the previous one?