

Correlated Trait Locus (CTL) mapping using the R Language for statistical computing

Danny Arends

Yang Li

Gudrun A. Brockmann

Ritsert C. Jansen

Robert W. Williams

Pjotr Prins



Overview

- * **Introduction**

- * Getting setup
 - * Text editor, R, R/CTL package
- * Correlated trait locus (CTL) mapping

- * **CTL mapping basics**

- * Mapping QTL and CTL

- * **Advanced CTL mapping**

- * Preparing data from GeneNetwork for CTL mapping
- * CTL mapping on BXD mice using GeneNetwork

Introduction

Correlated trait locus (CTL) mapping

Danny Arends

Yang Li

Gudrun A. Brockmann

Ritsert C. Jansen

Robert W. Williams

Pjotr Prins



Slides / Assignments / Answers

- * This presentation
 - * Slides, Assignments and Answers:

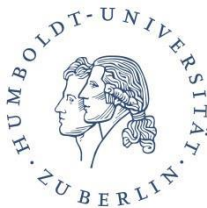
<https://www.dannyarends.nl/ctc/>

PDF slides with [links](#) to downloads

- * If you use the method, please cite our paper:

Arends et al, (2016), Correlation Trait Loci (CTL) mapping: phenotype network inference subject to genotype, Journal of Open Source Software, 1(6), 87, doi:10.21105/joss.00087

<http://joss.theoj.org/papers/440382846e5184c5ad875f8a53cf266d>



ERROR Please raise your hand

- * If something did not install / produced an error
 - * Raise you hand, some one will come and help you
 - * Do this ANY TIME something does not work



Install a good text editor

- * MS Windows

Notepad++

<https://notepad-plus-plus.org/download>

- * (Mac) OS X

TextWrangler (Get it from the app store / iTunes)

- * Linux

Use anything you want



Installing R

- * Minimal version > 3.3.0

- * MS Windows:

- <https://cran.r-project.org/bin/windows/base/>

- * (Mac) OS X:

- <https://cran.r-project.org/bin/macosx/>

- * Linux:

- <https://cran.r-project.org/bin/linux/>


- or install via your package manager



Start a new R document

- * Create a new folder to work inside
 - * Inside create a new text-file with the extension .R
- * Store all R commands in this file
 - * Comments by prefixing sentences with #

This is a comment and will be ignored

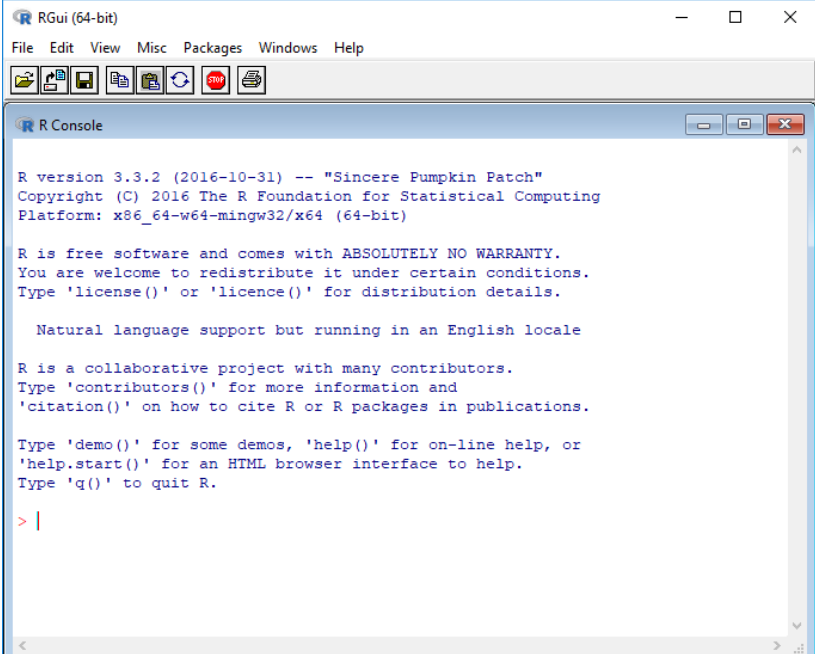
Name	Änderungsdatum	Typ	Größe
 workshop_code.R	5/28/2018 4:55 PM	R-Datei	0 KB



Install the CTL package

* Open up R, type:

```
install.packages("ctl")
```



The screenshot shows the RGui (64-bit) window with the R Console open. The console displays the following text:

```
R version 3.3.2 (2016-10-31) -- "Sincere Pumpkin Patch"  
Copyright (C) 2016 The R Foundation for Statistical Computing  
Platform: x86_64-w64-mingw32/x64 (64-bit)  
  
R is free software and comes with ABSOLUTELY NO WARRANTY.  
You are welcome to redistribute it under certain conditions.  
Type 'license()' or 'licence()' for distribution details.  
  
Natural language support but running in an English locale  
  
R is a collaborative project with many contributors.  
Type 'contributors()' for more information and  
'citation()' on how to cite R or R packages in publications.  
  
Type 'demo()' for some demos, 'help()' for on-line help, or  
'help.start()' for an HTML browser interface to help.  
Type 'q()' to quit R.  
  
> |
```



Install the CTL package

* Open up R, type:

```
install.packages("ctl")
```

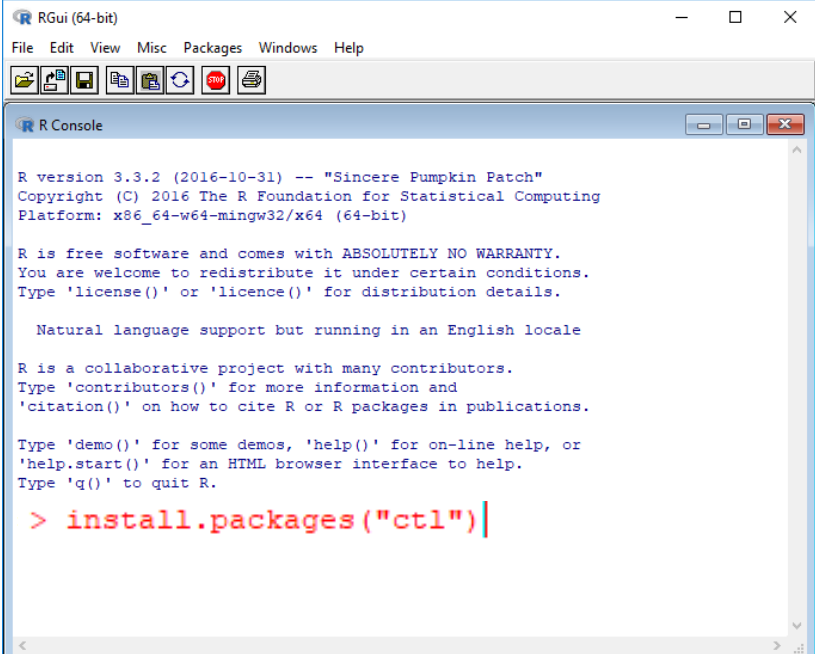
* Install from
o-Cloud
UK mirrors

HTTPS CRAN mirror

0-Cloud [https]
Algeria [https]
Australia (Canberra) [https]
Australia (Melbourne 1) [https]
Australia (Melbourne 2) [https]
Australia (Perth) [https]
Austria [https]
Belgium (Ghent) [https]
Brazil (PR) [https]
Brazil (RJ) [https]
Brazil (SP 1) [https]
Brazil (SP 2) [https]
Bulgaria [https]
Canada (BC) [https]
Canada (MB) [https]
Canada (NS) [https]
Chile 1 [https]
Chile 2 [https]
China (Beijing) [https]
China (Hefei) [https]
China (Guangzhou) [https]
China (Lanzhou) [https]
China (Shanghai 1) [https]
China (Shanghai 2) [https]

OK

Cancel



```
RGui (64-bit)
File Edit View Misc Packages Windows Help

R Console

R version 3.3.2 (2016-10-31) -- "Sincere Pumpkin Patch"
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> install.packages("ctl")
```



If everything went as planned

```
> install.packages("ctl")
Installing package into 'C:/Users/Arends/Documents/R/win-library/3.3'
(as 'lib' is unspecified)
--- Please select a CRAN mirror for use in this session ---
trying URL 'https://cloud.r-project.org/bin/windows/contrib/3.3/ctl_1.0.0-0.zip'
Content type 'application/zip' length 3906233 bytes (3.7 MB)
downloaded 3.7 MB

package 'ctl' successfully unpacked and MD5 sums checked
> |
```

- * Load package using the library function:

```
library("ctl")
```

```
> library("ctl")
Loading required package: MASS
Loading required package: qtl
> |
```



biomaRt

Query ENSEMBL from R

- * biomaRt is available from Bioconductor
- * Install biomaRt, type into R

```
source("http://bioconductor.org/biocLite.R")  
biocLite("biomaRt")
```

- * Load the biomaRt package

```
library("biomaRt")
```



All done

- * All set for CTL mapping



Cytoscape

- * Later on we'll create networks
- * Install Cytoscape (2.7.0) to visualize these networks

<http://www.cytoscape.org/>

Download Cytoscape 3.6.1

 for Windows (64 bit)

Java 8 will be automatically installed if not already present

Java 9 is not yet supported

Problems? [Read this page first](#)

[Release Notes](#)

[Other Platforms](#)

[Old Versions](#)



Introduction

- * QTL mapping
 - * Short introduction
 - * QTL mapping methods
 - * Limitations
- * CTL mapping
 - * Application of CTL mapping
 - * Introduction into the methodology

Quantitative trait locus (QTL)

Short introduction

A quantitative trait locus (QTL)

is a section of DNA (locus)

which is associated with

variation in a phenotype (quantitative trait)

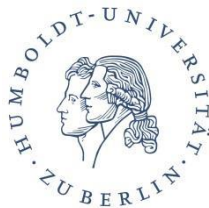
QTL mapping

Short introduction

- * To detect a QTL in a population
 - * Measure genotypes (e.g. SNPs)
 - * Phenotype of interest (e.g. Tail length)

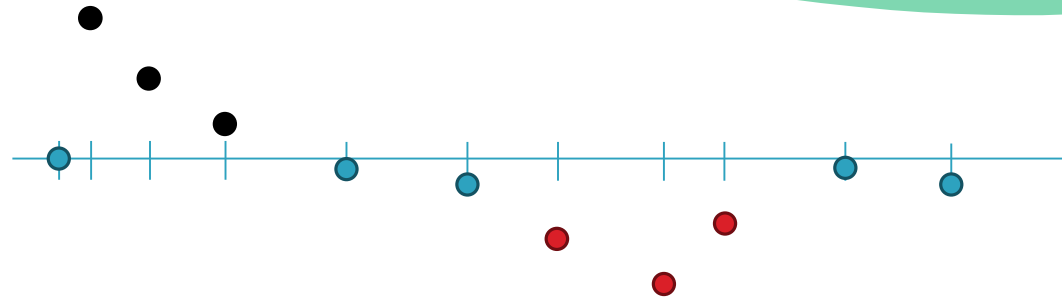
ID	Phenotype	Genotypes
Mouse 1	6.0	B A B B B A B A B A B A B B B A B A B A
Mouse 2	6.5	A A B A B A B B B B B A B B B A B A B A
Mouse 3	7.0	B A A B B B A A B B B A A B A A B B A A
Mouse 4	7.1	A A A B A A B B B A A A A B A A B B A A
Mouse 5	7.5	B A B A A A B B B A A B A A B A B A A B
Mouse 6	8.3	A B A A A A B A A A A B A A B A B A A B
Mouse 7	8.6	A B B A A B A A A B A A B B A B B A B
Mouse 8	8.9	A B A B B B B B A A A B A A A A B A A B
Mouse 9	9.1	B B A B A B A B A B A B A A A A B A A A
Mouse X	9.6	B B B B A A A A A B A B A A A A B A A A

Genetic Marker



QTL mapping

Short introduction



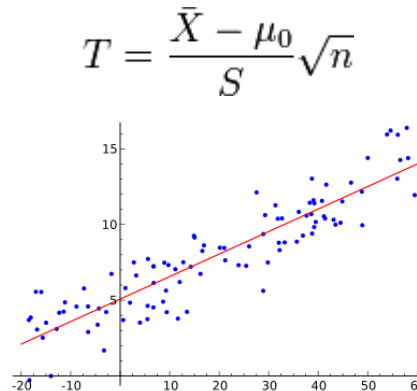
ID	Phenotype	Genotypes
Mouse 1	6.0	B A B B B A B A B A B B B A B A B A
Mouse 2	6.5	A A B A B A B B B B B A B B B A B A B A
Mouse 3	7.0	B A A B B B A A B B B A A B A A B B A A
Mouse 4	7.1	A A A B A A B B B A A A A B A A B B A A
Mouse 5	7.5	B A B A A A B B B A A B A A B A B A A B
Mouse 6	8.3	A B A A A A B A A A A B A A B A B A A B
Mouse 7	8.6	A B B A A B A A A B A A A B B A B B A B
Mouse 8	8.9	A B A B B B B B A A A B A A A A B A A B
Mouse 9	9.1	B B A B A B A B A B A B A A A A B A A A
Mouse X	9.6	B B B B A A A A A B A B A A A A B A A A

QTL mapping

Short introduction

- * Use statistics to assess the association of a marker

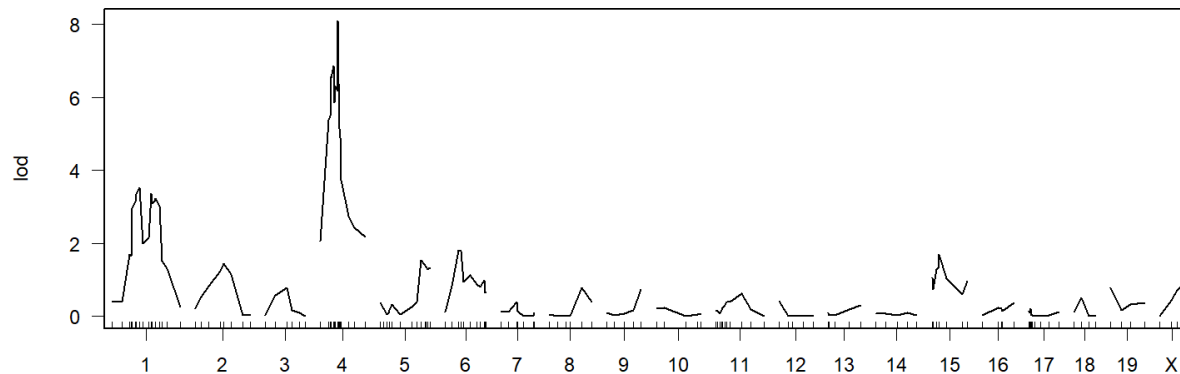
- * T-test
- * Anova
- * Regression



- * In QTL likelihoods are expressed as LOD score
 $-\log_{10}(\text{p-value})$

QTL mapping using R/qtl

- * Likelihoods are plotted on the chromosomes



- * In R we can use R/qtl for QTL mapping:

```
library(qtl)           # Installed by the ctl package
data(hyper)           # Load the dataset
plot(scanone(hyper))  # Scan for QTL and plot the results
```

QTL mapping

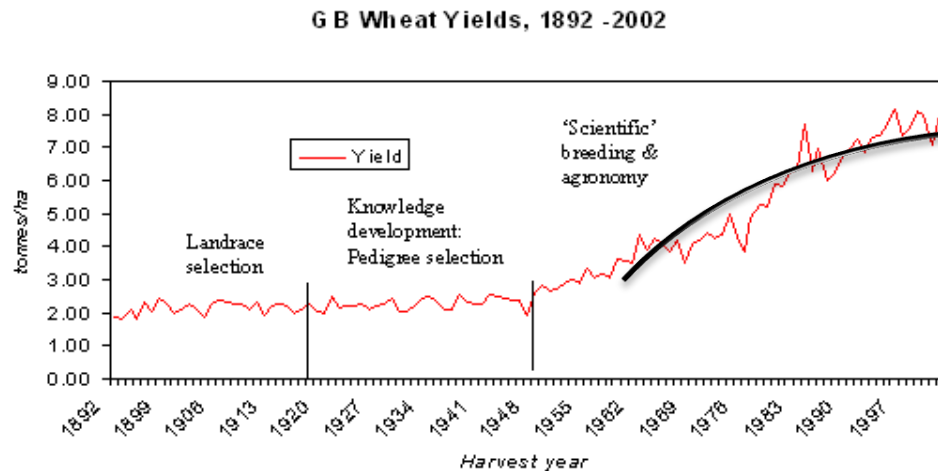
Limitations

- * Considers information of one phenotype at a time
- * Requires the phenotype to show significant differences
- * Highly correlated phenotypes result in similar QTL profiles

QTL mapping

Limitations

- * Imagine two 'linked' phenotypes
 - * Susceptibility to infection & Plant yield
- * Bigger yields = higher susceptibility to infection
- * In plants: ***Yield plateau***



CTL mapping

A correlated trait locus (CTL)

is a section of DNA (locus)

which is associated with

differences in correlation between phenotypes



CTL mapping

- * CTL is a method similar to QTL mapping
 - * Multi-phenotype
 - * Identify genetic regions where there is a difference in phenotype-phenotype correlation
 - * Conditional on the genotype of a genetic marker
- * Unbiased, data driven (no prior information)

Applications of CTL mapping

- * Classical selection in breeding to improve economically interesting phenotypes
 - * Select for beneficial CTL loci, similar to QTL
 - * Break linkage between phenotypes
- * Combined with QTL data
 - * build a phenotype x phenotype connection network

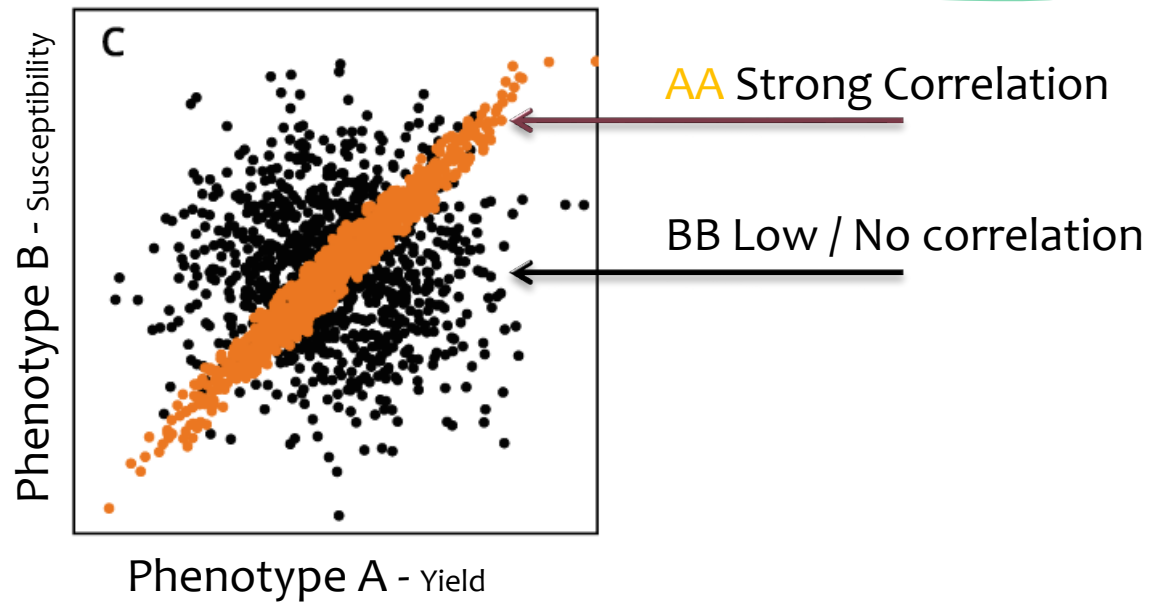
Example

Yield Plateau

- * Both phenotypes (yield, susceptibility) will show similar QTL profiles because of their overall correlation
 - * When we select for a high yield genotype, this unfortunately also leads to increased susceptibility
- * CTL mapping finds loci where correlation between yield and susceptibility is lost
 - * We use CTL information identify genotypes able to ‘break’ the correlation between yield and susceptibility

Example

Yield Plateau



Which genotype **AA** or **BB** should we breed ?

Example

Yield Plateau

- * Select a CTL genotype to ‘unlink’ the two phenotypes
- * The next generation
 - * Should show a **decreased correlation** between phenotypes
 - * Allows selection of high yield without increasing susceptibility
- * Select individuals on QTL information as normal

CTL mapping

Methodology

- * To explain the method, I use recombinant inbred lines
 - * CTL package handles other more complex crosses
- * Recombinant inbred lines
 - * Four phenotypes measured
 - * Six Genetic markers
 - * Two alleles are possible: AA and BB

CTL mapping

Methodology

- * Select a phenotype (**p1**)
 - * Select a genetic marker
 - * Split the individuals in two groups by genotype (**AA**, **BB**)
 - * Calculate correlation for both the **AA** and the **BB** genotype
- p1 x Phenotypes** correlation vector

cor(**P1**, Px in **AA**)

cor(**P1**, Px in **BB**)

[p1, p2, p3, p4]

[1.0, 0.1, 0.5, 0.8]

[1.0, 0.2, 0.3, 0.1]

CTL mapping

Methodology

- * QTL effect size
 - * Difference in mean between AA and BB
- * The CTL effect size
 - * Difference in correlation between AA and BB

	[p1, p2, p3, p4]
cor(P1, Px in AA)	[1.0, 0.1, 0.5, 0.8]
cor(P1, Px in BB)	[1.0, 0.2, 0.3, 0.1]
Difference	[0.0, 0.1, 0.2, 0.7]



CTL mapping

Methodology

- * Repeat this calculation for every genetic marker
- * Multiple difference vectors for our **phenotype**

Result from the last slide

	m1	m2	m3	m4	m5	m6
p1	[0.0,	0.0,	0.0,	0.0,	0.0,	0.0]
p2	[0.1,	0.0,	0.1,	0.1,	0.1,	0.0]
p3	[0.2,	0.3,	0.5,	0.6,	0.6,	0.4]
p4	[0.7,	0.6,	0.6,	0.4,	0.2,	0.1]

CTL mapping

Methodology

- * Repeat this calculation for every genetic marker
- * Multiple difference vectors for our **phenotype**

Note: Mapping p1 against p1, always yields a difference of 0

	m1	m2	m3	m4	m5	m6
p1	0.0	0.0	0.0	0.0	0.0	0.0
p2	0.1	0.0	0.1	0.1	0.1	0.0
p3	0.2	0.3	0.5	0.6	0.6	0.4
p4	0.7	0.6	0.6	0.4	0.2	0.1

Intermezzo

permutation

- * Repeat 10.000 times
 - * Break the link between genotype and phenotype
 - * Assign genotypes at random to individuals
 - * Redo the whole analysis
 - * Remember the maximum observed score
- * Make a distribution out of 10.000 scores
- * Find 5% and 1% threshold values for significance

CTL mapping

Methodology

- * Use 10.000+ permutations to assign significance
- * Or use direct calculation of p-values using mathematics
- * Convert differences to p-values
- * Convert p-values to LOD ($-\log_{10}(p\text{-value})$)

CTL mapping

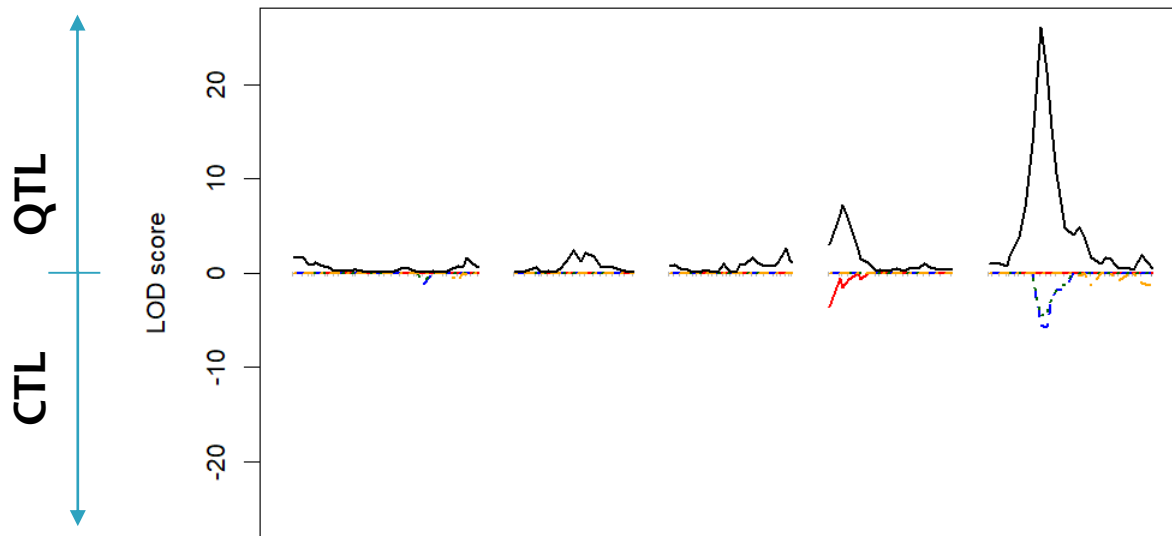
Methodology

- * Convert the values to $-\log_{10}(p\text{-value})$
- * Perform QTL mapping for p1
 - * For p1: 4 vectors of LOD scores from CTL mapping
1 vector of LOD scores from QTL mapping

	m1	m2	m3	m4	m5	m6
p1	0.0	0.0	0.0	0.0	0.0	0.0
p2	0.5	0.0	0.4	0.5	0.2	0.0
p3	1.0	1.5	4.9	6.0	6.0	4.5
p4	7.1	6.2	6.2	5.1	2.2	0.5
QTL	0.9	2.5	5.6	7.1	3.3	1.1

CTL mapping visualization

- * QTL curve of our **p1**
- * CTL curves our **p1** versus the other phenotypes



CTL and Network inference

- * CTL are the result of QTL in upstream phenotypes
 - * CTL identifies phenotype(s) in the same network
- * Regulatory arrows between phenotype levels
 - * CTL mapping of a classical phenotypes against gene expression traits
- * CTL can be used to detect meaningful biological causal connections between phenotypes

CTL example

- * 162 Recombinant Inbred Lines *A. thaliana*
 - * 5 chromosomes characterized at 69 markers
 - * 24 characterized metabolites
 - * I focus on 3/24 metabolites
- * Not the best example data
 - * Metabolites (needs a 2 part model)
 - * Many 0's in the data scewing the correlation

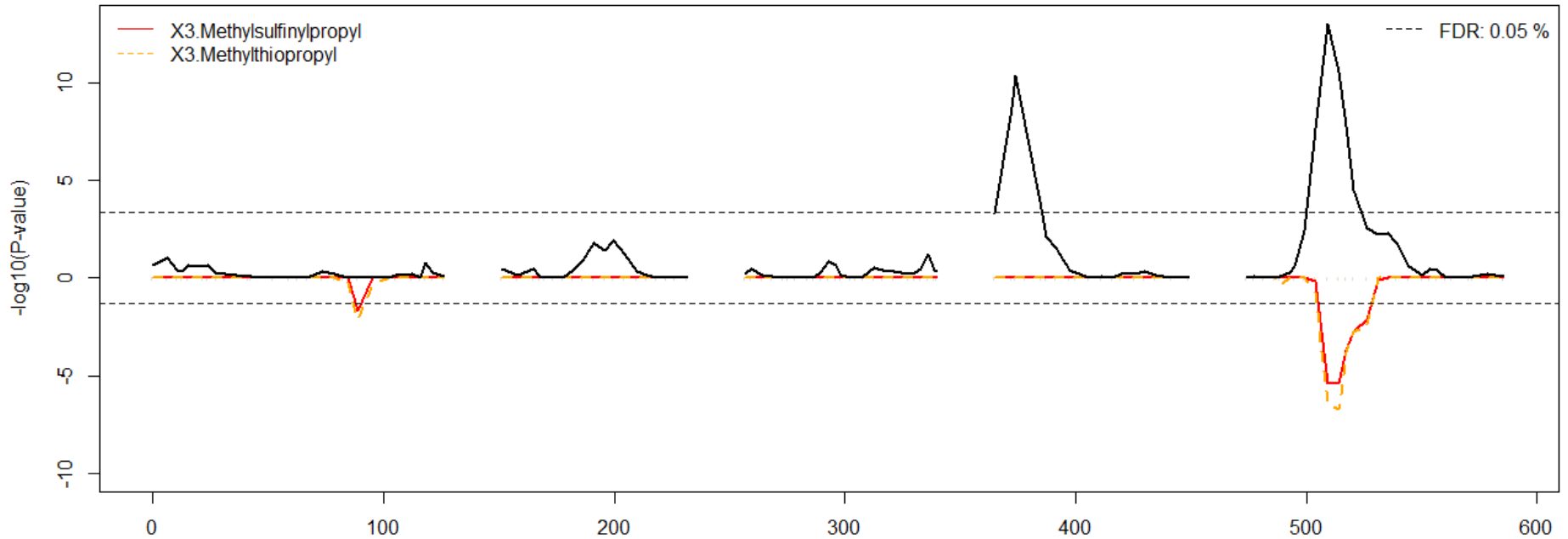
CTL example

- * The metabolites here are known to be in a linear pathway:
 - * X₃.Hydroxypropyl
 - * X₃.Methylsulfinylpropyl
 - * X₃.Methylthiopropyl
- * Major regulator of this pathway on chromosome 5

QTL & CTL

X3.Hydroxypropyl

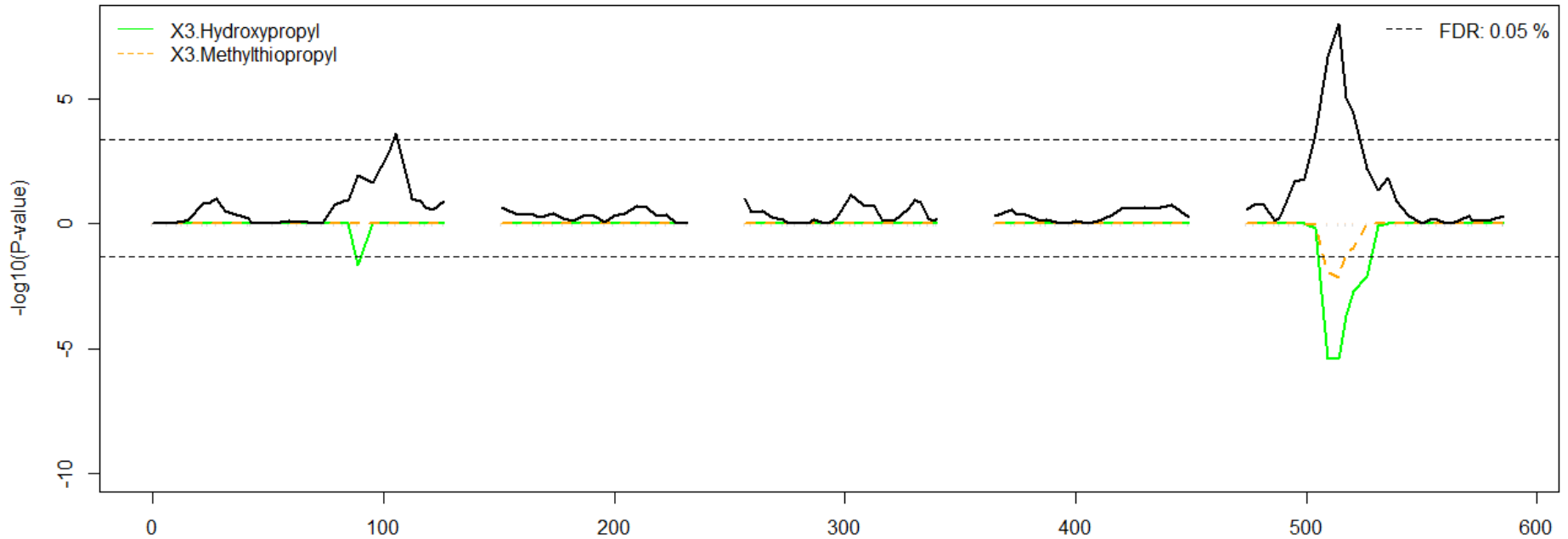
X3.Hydroxypropyl



QTL & CTL

X₃.Methylsulfinylpropyl

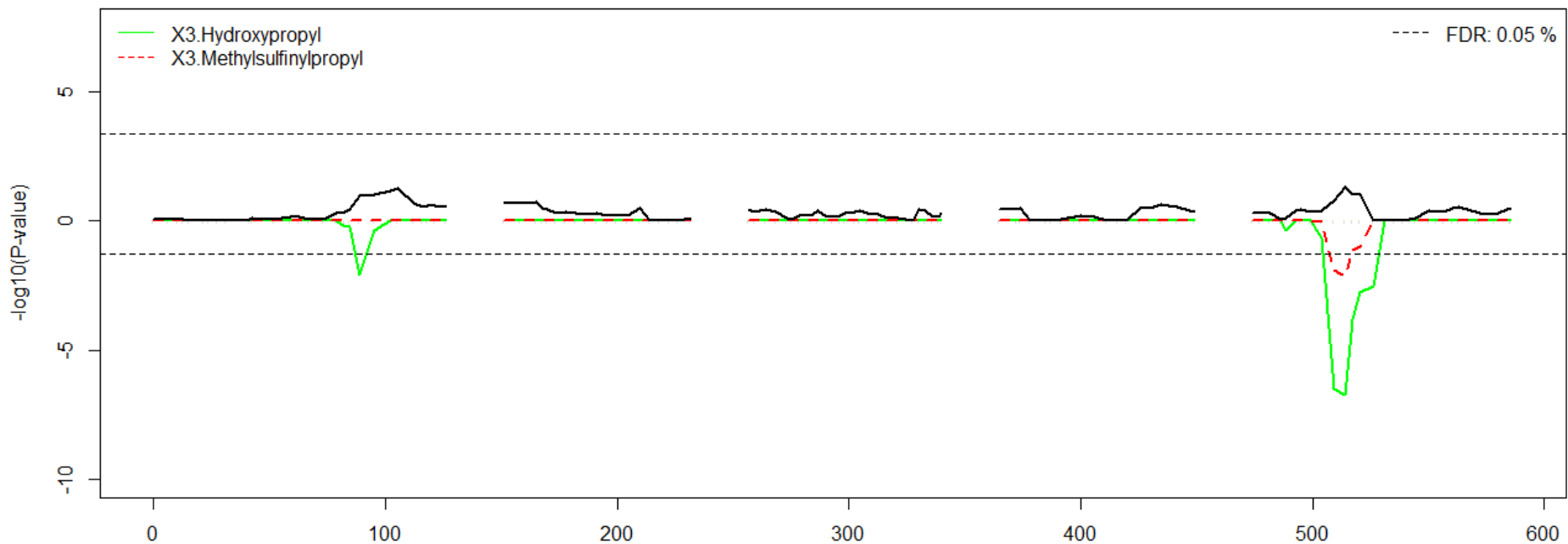
X₃.Methylsulfinylpropyl



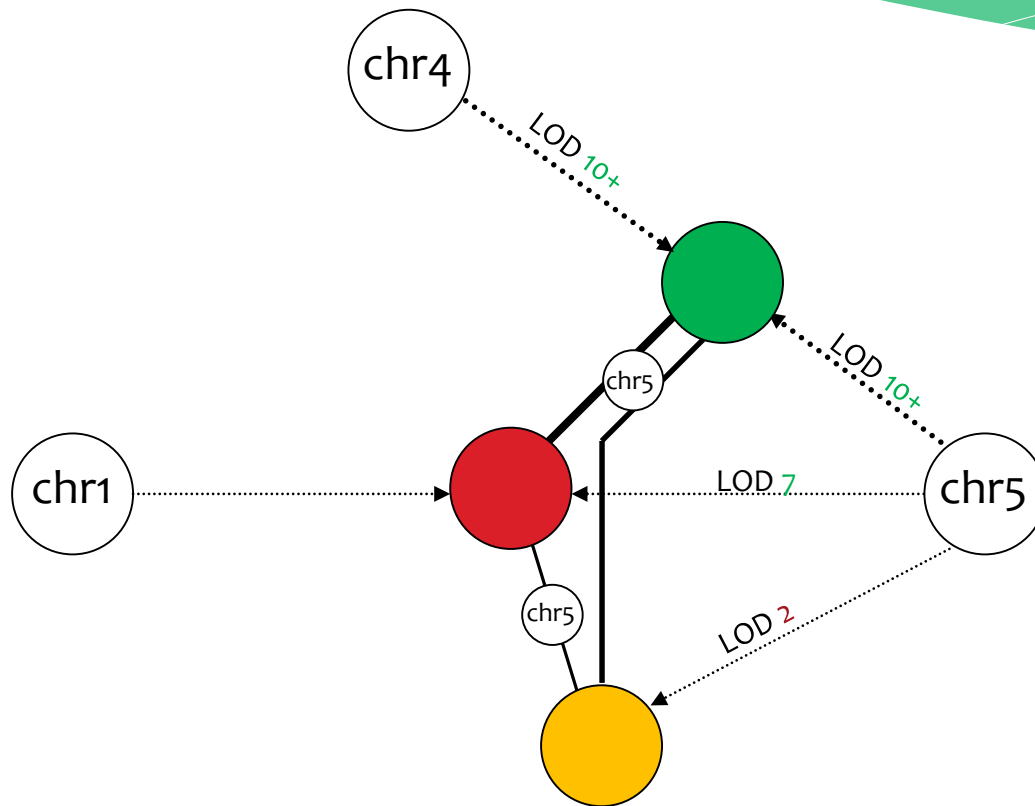
QTL & CTL

X₃.Methylthiopropyl

X₃.Methylthiopropyl



Reconstructed Network



X3.Hydroxypropyl
X3.Methylsulfinylpropyl
X3.Methylthiopropyl

← QTL
CTL

End of the introduction

- * Session overview
 - * Installed: Text editor, R, ctl, and biomaRt
 - * Yield plateau / CTL methodology
 - * CTL example in *A. Thaliana*

CTL mapping basics

Danny Arends

Yang Li

Gudrun A. Brockmann

Ritsert C. Jansen

Robert W. Williams

Pjotr Prins



Overview

- * Goals

- * Map CTLs within a Yeast gene expression data set
- * Get familiar with the R/ctl package functions
- * Find significant CTL
 - * Create several plots to show/investigate
- * Annotate the probes to genes
- * Use Cytoscape to visualize the network

CTL mapping basics

- * CTL package functions

CTLscan - Main function to scan for CTL

getCorrelations - Get effect and sample size

plot - plot CTL objects

CTLsignificant - Get all significant interactions from a genome-wide CTLscan.



Set a working directory

- * Using the `setwd` command, R can move into a directory on your hard drive
 - * Change the backslashes (`\`) in the path to (`/`)

```
setwd("D:/CTCworkshop")
```

- * New files are now be stored in this location

Example data

- * Load the example data available with the package

```
data(yeast.brem)
```

The example data consists of 3 matrices: genotypes, phenotypes, and a genetic map. Show a small subset of the data using the *head* function

```
head(yeast.brem$genotypes)  
head(yeast.brem$phenotypes)  
head(yeast.brem$map)
```

Data description

- * *Saccharomyces cerevisiae* recombinant inbred lines
 - * 109 lines
 - * 301 gene expression probes
 - * 282 genetic markers on 16 chromosomes

First let's get rid of having to type: "yeast.brem" multiple times

```
phenotypes = yeast.brem$phenotypes  
genotypes = yeast.brem$genotypes  
map = yeast.brem$map
```



First CTL scan

- * Let's scan the fourth probe on the array against the other probes

```
ctlres = CTLscan(genotypes, phenotypes, phenocol = 4)
```

- * The resulting object is a list, since we only scanned 1 phenotype it should have length 1

```
length(ctlres)
```

```
# Should be 1 !
```



A look inside the CTL object

- * Summary of the whole object

```
ctlres  
  
> ctlres  
CTLObject summary  
  
- Number of scanned phenotypes: 1
```

- * Summary of an individual scan

```
ctlres[[1]]  
  
> ctlres[[1]]  
CTLscan summary  
  
- Number of background phenotypes: 301  
- Number of markers: 282  
- Number of permutations: 0  
Found 3 significant CTLs
```



A look inside the CTL object

- * Use a \$ sign after the object and [tab] to autocomplete the fields inside the object

```
ctlres[[1]]$  
> ctlres[[1]]$  
ctlres[[1]]$qtl   ctlres[[1]]$dcor   ctlres[[1]]$perms   ctlres[[1]]$ctl
```

- * 4 objects inside: qtl, dcor, perms, and ctl

A look inside the CTL object

- * The 4 objects inside:
 - qtl**: Vector of LOD scores for QTL likelihood
 - dcor**: Matrix of Z scores for each trait at each marker
 - ctl**: Matrix of LOD scores for CTL likelihood
 - perms**: Vector of maximum scores obtained during permutations
- * Use the head function to see parts of the objects

```
head(ctlres[[1]]$ctl)
```



Plot the results

- * The default plot function can plot the ctl object

```
plot(ctlres)
```

- * The mapinfo parameter provides the genetic map

```
plot(ctlres, mapinfo = map)
```

Try plotting different visualizations using the type parameter:
"barplot", "gwas", and "line"



Options for CTLscan

- * **strategy:** How to map CTLs
 - * "Exact", "Full", or "Pairwise"
- * **conditions:** Specify a conditions vector to be used in QTL mapping
- * **qtls:** Specify QTLs calculated by another tool (R/qtl, Plink, etc)
- * **parametric:** use either non-parametric testing (Spearman) or parametric testing (Pearson)
- * **ncores:** Number of CPU cores to use
- * **adjust:** Adjust p-values for multiple testing ?

CTLsignificant & getCorrelations

- * To get an overview of significant effects

```
signres = CTLsignificant(ctlres)
```

- * A_06_P4255 shows a CTL with A_06_P4854 on three markers on chromosome 1, inspect the correlation change by using the *getCorrelations* function

```
allcors = getCorrelations(genotypes, phenotypes,  
                          phenocol = 4, marker = "R006W_131")
```



Put the information together

- * Spearman correlation differences A_06_P4255 and A_06_P4854

```
allcors$correlations["A_06_P4854",]
```

- * Use: parametric = TRUE for Pearson correlations

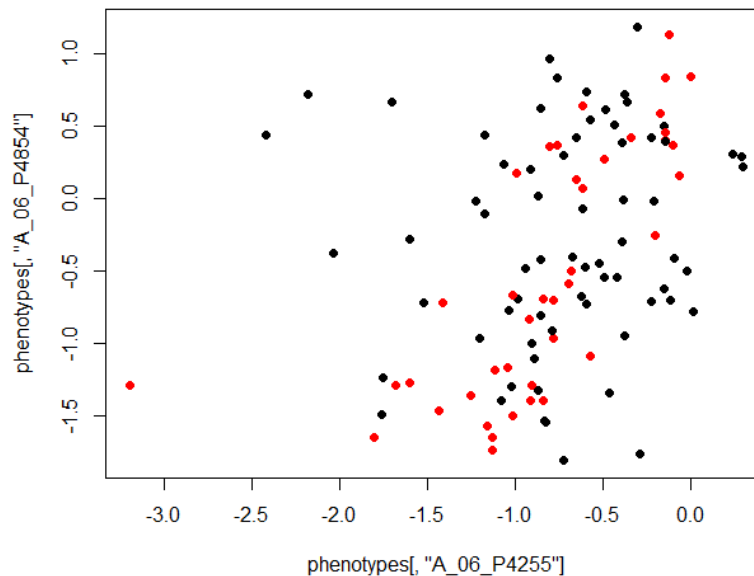
```
allcors = getCorrelations(genotypes, phenotypes, phenocol = 4,  
                          marker = "R006W_131", parametric=TRUE)  
allcors$correlations["A_06_P4854",]
```



Show the correlation

- * Plot the data points using the marker as color

```
plot(phenotypes[, "A_06_P4255"], phenotypes[, "A_06_P4854"],  
     col = genotypes[, "R006W_131"], pch=19)
```



Multi trait scans

- * Using nthreads we can speed up computation when mapping all phenotypes:

```
ctlall = CTLscan(genotypes, phenotypes, nthreads = 3)
```

- * Adjust the nthreads to be one lower than the actual number of CPU cores in your system
 - * not hyper-threaded cores

Plot a multitrait scan

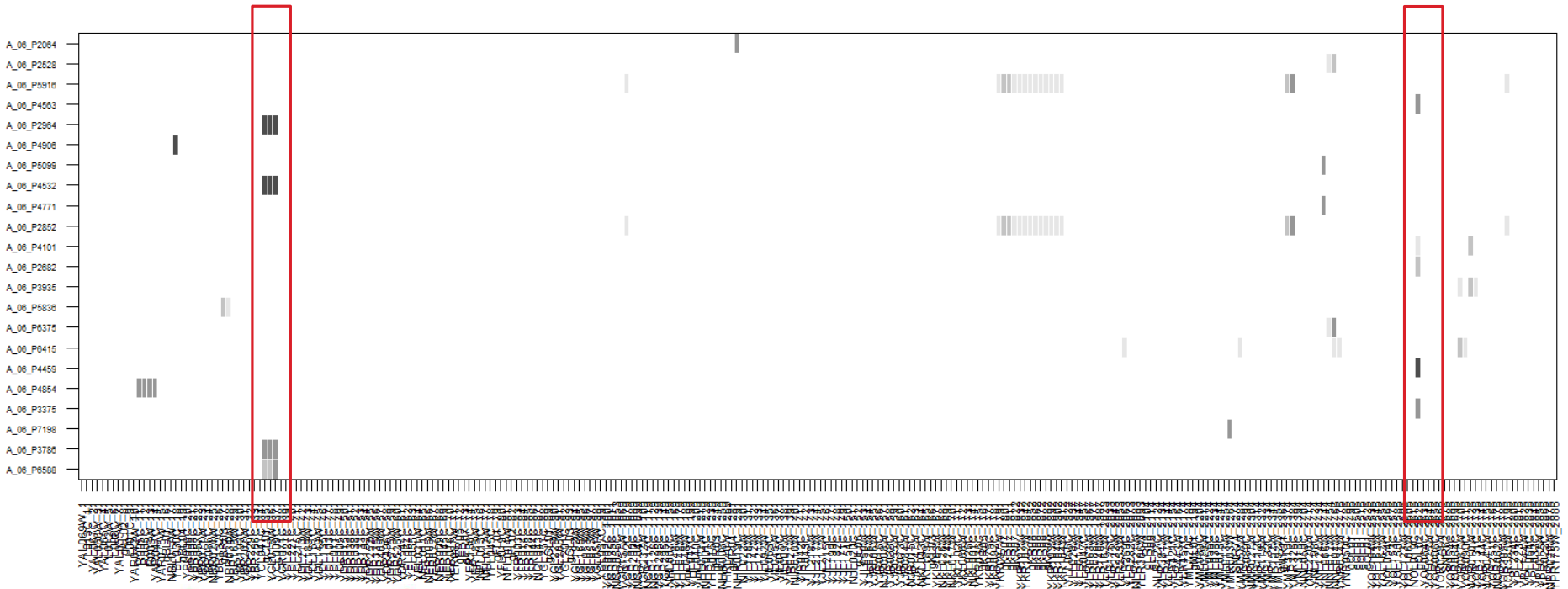
- * Check if all phenotypes got scanned:

```
length(ctlall)          # Should be 301
```

- * Plot an overview picture of all significant CTL

```
plot(ctlall, significance = 0.01) # or 0.05
```

Plot of a multitrait scan



Significant CTL / Hotspots

- * List all significant CTLs found during the scan

```
signCTL = CTLsignificant(ctlall)
```

- * Find how many CTLs exist per marker

```
table(signCTL[, "marker"])
```

- * In our example marker "gOLo2_2626" has 8 CTL



gOL02_2626

- * Where is this marker located ?

```
map["gOL02_2626",]
```

- * Which probe expressions are affected ?

```
signCTL[which(signCTL[, "marker"] == "gOL02_2626"),]
```

Know your IDs

- * Find the IDs mapping to gOL02_2626

```
signCTLids    = CTLsignificant(ctlall, what = "ids")
gOL02mapping  = which(signCTL[, "marker"] == "gOL02_2626")
gOL02sign     = signCTLids[gOL02mapping, ]
```

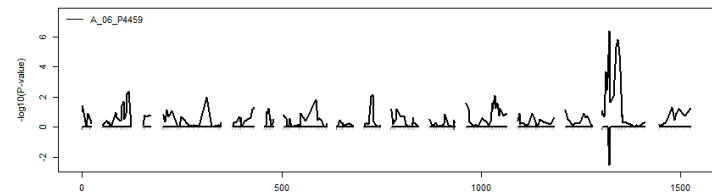
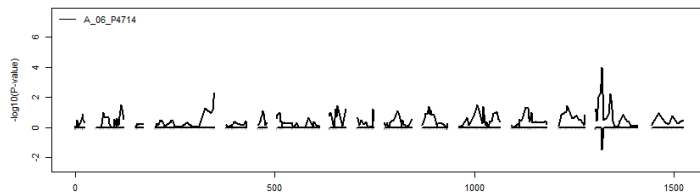
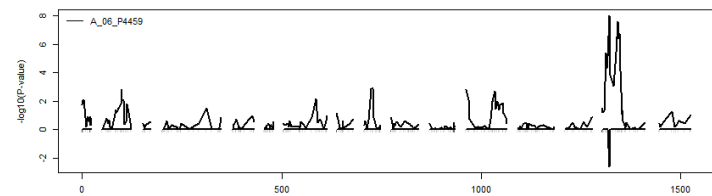
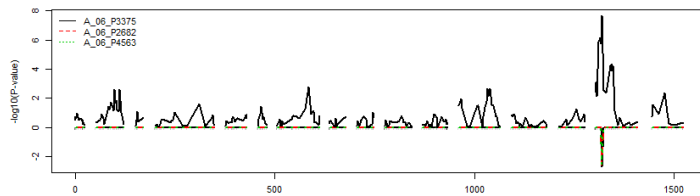
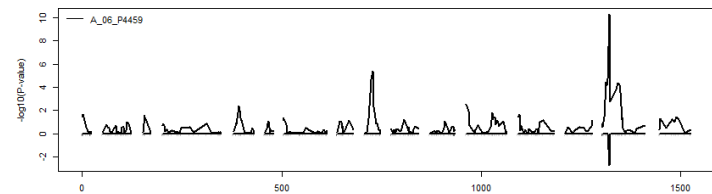
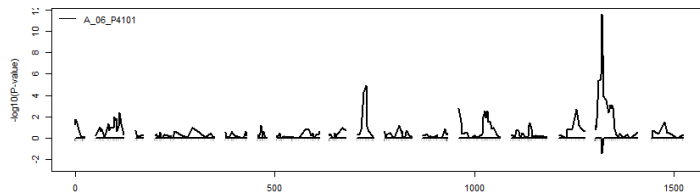
- * Plot all profiles mapping to gOL02_2626

```
par(mfrow=c(3, 2), mai = c(1, 1, 0, 0)) # Plot 3x2 with margins

for(x in unique(gOL02sign[,1])){
  plot(ctlall[[x]], mapinfo = map, t = "lineplot")
}
```



6 probes show CTL at gOLo2_2626



Download the annotation

- * Get the annotation from:

<https://www.dannyarends.nl/CTC2018/annotation.txt>

- * Load in the probe annotation with read.table

```
annotation <- read.table(file = "annotation.txt",  
                        sep = "\t",  
                        header = FALSE,  
                        row.names = 1)
```

```
head(annotation) # show the top of the annotation
```



Which genes are mapping to gOL02_2626

- * Extract the names of the probes

```
probenames <- as.character(unique(signCTL[gOL02mapping, 1]))
```

- * Use the annotation, to annotate them

```
annotation[probenames,]
```

Biology ?

- * We find a cluster of six genes showing CTLs at this marker, four of which are known

Ecm4 (AKA: Gto2), Gto1, Gip2, Pry1

- * and 2 additional hypothetical proteins

YKLo50C, NNR2

"Saccharomyces cerevisiae cells contain three omega-class glutathione transferases with glutaredoxin activity (Gto1, Gto2, and Gto3)" [10.1128/EC.00216-06](#)

"Glutathione transferases have important roles in cellular detoxification" [10.1080/13102818.2007.10817472](#)

"Pry1 may be involved in detoxification of hydrophobic compounds" [10.1073/pnas.1209086109](#)

"Gip2 regulates the aurofusarin biosynthetic" [10.1128/AEM.72.2.1645-1652.2006](#) & "aurofusarin is a homodimeric naphthoquinone" & "The second mechanism of naphthoquinone toxicity involves redox cycling, with generation of "active oxygen" species."

"Once oxidized, glutathione can be reduced back by glutathione reductase, using NADPH as an electron donor." [10.1021/pr4001948](#) (NNR2 = NADHX dehydratase)

Some evidence that these genes have something to do with detoxification of yeast cells, and protection against active oxygen" species. However it is unclear HOW they are related

We do know that they share a QTL at marker gOLo2_2626, and these genes show correlation loss/gain at this marker



Networks

- * CTL mapping results on multiple traits can be used to construct networks, by using the CTLnetwork function

```
CTLnetwork(ctlall, mapinfo = map,  
           file="network", significance = 0.01)
```

```
> CTLnetwork(ctlall, mapinfo = map, file="network", significance = 0.01)  
Found 36 significant CTLs  
NETWORK.SIF  
NODE.DESCRPTION
```



Cytoscape

Import Network and Edge Attributes from Table


Import Network from Table

Data Sources

Input File: file:///D:/CTCworkshop/ctlnetnetwork.sif Select File(s)

Interaction Definition

Source Interaction: Column 1 Interaction Type: Column 2 Target Interaction: Column 3

 Columns in BLUE will be loaded as EDGE ATTRIBUTES.

Advanced

Show Text File Import Options

Preview

Text File: ctlnetnetwork.sif

Left Click: Enable/Disable Column, Right Click: Edit Column

Column 1	Column 2	Column 3	Column 4	Column 5
A 06 P6588	QTL	YCL047C 341	QTL	2.109977
A 06 P6588	QTL	gCL01 351	QTL	15.04908
A 06 P6588	QTL	YCL018W 361	QTL	15.30008
A 06 P6588	QTL	YCL009C 371	QTL	13.2185
A 06 P6588	QTL	NFL012W 771	QTL	2.022834
A 06 P6588	CTL 11 259	gCL01 351	CTL	2.234957
gCL01 351	CTL 11 259	A 06 P2964	CTL	2.234957
A 06 P6588	QTL	YCL047C 341	QTL	2.109977
A 06 P6588	QTL	gCL01 351	QTL	15.04908

Import Cancel

* File ->

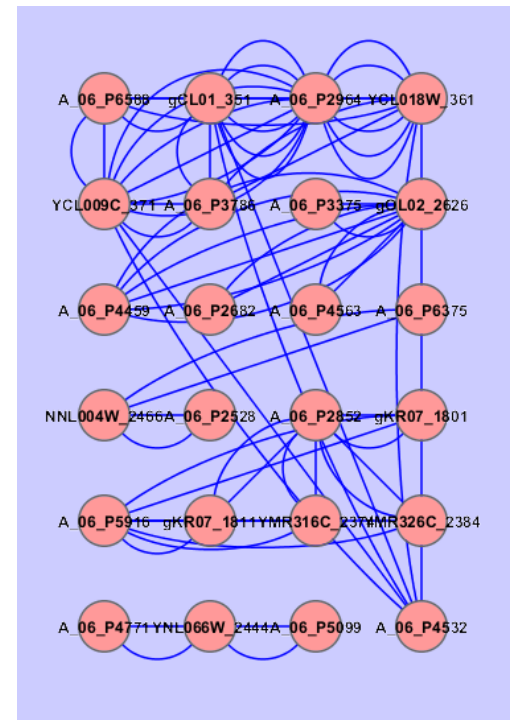
* Import ->

* Network
from Table



Network

- * Also load in the node properties
- * File ->
 - * import ->
 - * attributes from table
- * Default layout and style do not look very pretty
- * We can use the VisMapper tab to beautify the visualization



Small demo on Cytoscape

- * What I will quickly show
 - * Load the node and sif files
 - * Change the layout
 - * Change the look n feel
- * Optional things (probably won't show them)
 - * Probe names to gene-names
 - * Load the `ctlnetwork.nodes`
 - * Add the hyperlinks to the node



Session overview

- * Introduced the following functions
 - * CTLscan – scan for CTL
 - * getCorrelations – find the correlation differences
 - * Different plot routines
 - * CTLsignificant – get a list of significant CTLs
 - * CTLnetwork – Creates a network file to be used with Cytoscape
- * Basic introduction into Cytoscape
 - * Very powerful
 - * A LOT of options available

Advanced CTL mapping

Using BXD data from Genenetwork

Danny Arends

Yang Li

Gudrun A. Brockmann

Ritsert C. Jansen

Robert W. Williams

Pjotr Prins

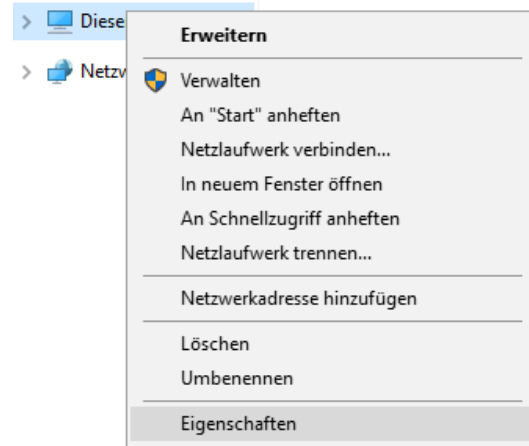


Information

- * For this section, we need the latest version of the `ctl` package from Github
 - * Slightly different from the CRAN version
- * Windows users need to install Rtools first
 - * This is not needed for linux

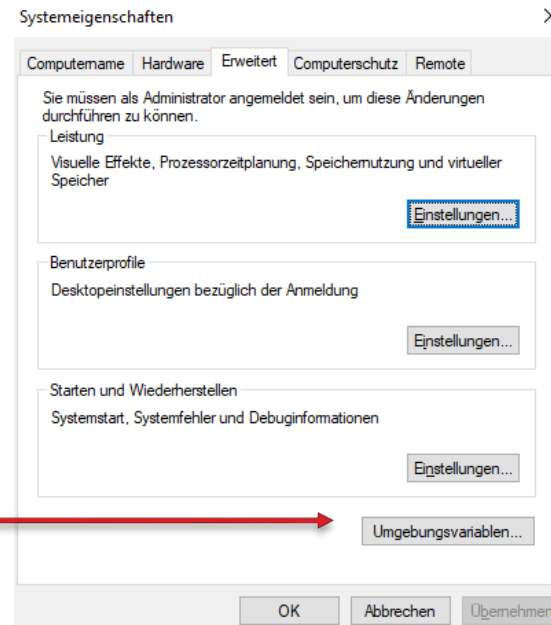
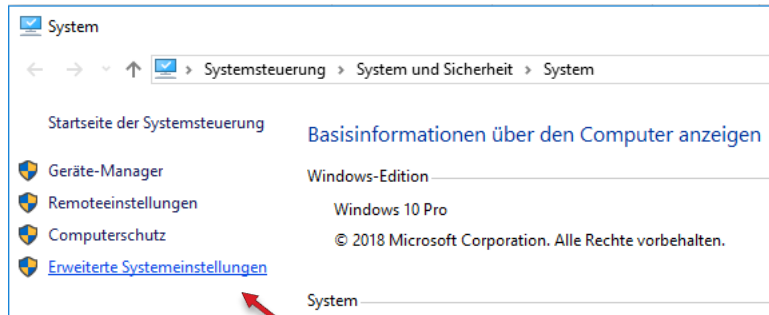
Install Rtools (Windows only)

- * Rtools is the compiler for MS Windows
- * Download it from:
 - * <https://cran.r-project.org/bin/windows/Rtools/>
- * After downloading add Rtools/ to your path
- * Right click My Computer
 - * Select Properties



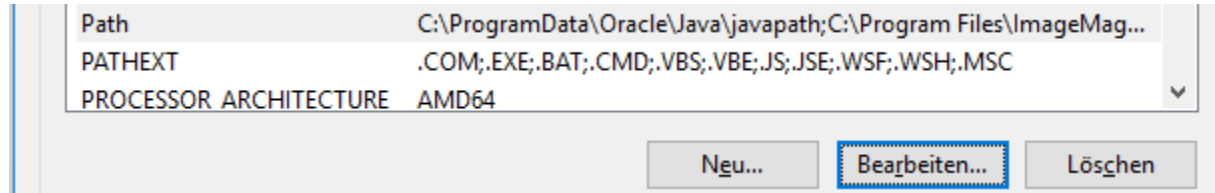
Install Rtools (Windows only)

- * Go to “Advanced system setting”, then to “Environmental variables”



Install Rtools (Windows only)

- * In “system variables”, select the Path line, after that click “edit”,



- * Add a the Rtools package to the Path variable, by clicking “new” type the location where Rtools was installed followed by \bin:
- * **c:\Rtools\bin**



Turn it on an off again

- * Make sure to **close** and **reopen** R
 - * After adding Rtools to your path



Install the CTL package from Github

- * Load the devtools library

```
install.packages("devtools")
```

- * Install the CTL package

```
library(devtools)  
install_github("DannyArends/CTLmapping", subdir="Rctl")
```

Advanced CTL mapping

- * BXD data is obtained from GeneNetwork

<http://www.genenetwork.org>

GeneNetwork

University of Tennessee: www.genenetwork.org

Use GeneNetwork 2

[Home](#) | [Search](#) | [Help](#) | [News](#) | [References](#) | [Policies](#) | [Links](#)

Select and Search

Species:

Group: [Info](#)

Type:

Data Set: [Info](#)

Databases marked with ** suffix are not public yet.
Access requires [user login](#).

Get Any:

Enter terms, genes, ID numbers in the **Get Any** field.
Use * or ? wildcards (Cyp*a?, synap*
Use **Combined** for terms such as *tyrosine kinase*.

Combined:

[Search](#) [Make Default](#) [Advanced Search](#)

GeneNetwork [Intro](#) [Search](#) [Submit Trait](#) [Collections](#) [Help](#) [News](#) [References](#) [Po](#)

Genes / Molecules [Search All](#)

Select and search

Species: [Make Default](#)

Group:

Type:

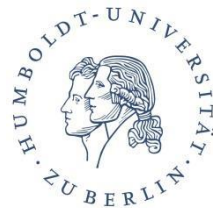
Dataset: [Info](#)

Get Any:

Enter terms, genes, ID numbers in the **Search** field.
Use * or ? wildcards (Cyp*a?, synap*
Use **quotes** for terms such as "tyrosine kinase".

Combined:

[Search](#)



BXD vantages

- * Derived by crossing C57BL/6J (B6) and DBA/2J (D2)
- * Inbred progeny for 20 or more generations
- * 40-years of phenotypes
- * Sequencing data on both parents (and the RI)
- * Closely related sub-strains
- * Epoch Differences among BXD strains
- * Many datasets have low number off strains

Mapping in BXD

- * We'll investigate GN111, the info can be found here:

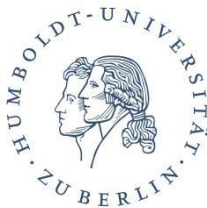
http://www.genenetwork.org/webqtl/main.py?FormID=sharinginfo&GN_AccessionId=111

- * **Hippocampus Consortium M430v2**

Summary:

MOST HIGHLY RECOMMENDED DATA SET (**Overall et al., 2009**): The Hip diverse strains of mice including 67 BXD recombinant inbred strains, 13 C

The hippocampus is an important and intriguing part of the forebrain that schizophrenia. Unlike most other parts of the brain, the hippocampus cont (Kempermann, **2005**). This genetic analysis of transcript expression in th described in the acknowledgments section.



Download the genotype data




- * <http://www.genenetwork.org/genotypes/BXD.geno>
- * Right-click: save as

```
# File name: BXD_Final_2017_7324_markers.geno
# Metadata: Please retain this header information with file.
# Data Source and Contact: GeneNetwork at http://www.genenetwork.org/webqtl/main.py?FormID=sharinginfo&GN_Acces
# Citation: For information on the BXD family of strains please see http://www.genenetwork.org/reference.html a
# Date Modified: Jan 19, 2017: Danny Arends computed BXD cM values and recombinations between markers. Rob W. W
# Coordinates of Markers and Assembly: Megabase and basepair positions of markers before 2017 used mm9 NCBI Bui
# Genotypes: This file provides consensus genotypes for 198 BXD strains and for the two progenitor strains and
# Material and Cases: 155 BXD strains and substrains are available as live stock from JAX or UTHSC (Jan 2017).
# Breeding: BXD1 to BXD30 generated by BA Taylor at the Jackson Laboratory starting in 1971 from F2 stock. BXD3
# Errors and Corrections: This file contains genotyping errors and imprecision in the locations of recombinatic
# cM position: cM positions were estimated by Danny Arends, Dec 2016. One recombination on both chromosomes (e.
# Acknowledgments: We thank Lu Lu for generating many of these strains (BXD43 and higher). We thank Casey Bohl,
# Funding: This work and GeneNetwork have been funded by The UTHSC Center for Integrative and Translational Ge
# Column Heads: Chr = chromosome, Locus=marker name, Mb_mm9= megabase position mouse genome assembly mm9, cM_Cc
@name:BXD
@type:riset
@mat:B
@pat:D
@het:H
@unk:U
Chr>Locus>cM>Mb>BXD1>BXD2>BXD5>BXD6>BXD8>BXD9>BXD11>BXD12>BXD13>BXD14>BXD15>BXD16>BXD18>BXD19>BXD20>BXI
1>rs31443144>1.50>3.010274>B>B>D>D>D>B>B>D>B>B>D>B>D>D>D>B>B>B>D>B>D>D>B>B>B>B>B>B>B>B>B>D>B>D>B>B>D>B>E
1>rs6269442>1.50>3.492195>B>B>D>D>D>B>B>D>B>B>D>B>D>D>D>B>B>B>D>B>D>D>B>B>B>B>B>B>B>B>B>D>B>D>B>B>D>B>E
1>rs32285189>1.63>3.511204>B>B>D>D>D>B>B>D>B>B>D>B>D>D>D>B>B>B>D>B>D>D>B>B>B>B>B>B>B>B>B>D>B>D>B>B>D>B>E
1>rs258367496>1.63>3.659804>B>B>D>D>D>B>B>D>B>B>D>B>D>D>D>B>B>B>D>B>D>D>B>B>B>B>B>B>B>B>B>D>B>D>B>B>D>B>E
1>rs32430919>1.75>3.777023>B>B>D>D>D>B>B>D>B>B>D>B>D>D>D>B>B>B>D>B>D>D>B>B>B>B>B>B>B>B>B>D>B>D>B>B>D>B>E
```



Download the phenotype data

- * <http://datafiles.genenetwork.org/download/GN111/>
- * Right-click: save as

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 GN111 MeanDataAnnotated rev081815.txt	20-Aug-2015 18:10	41M	
 GN111 StdError rev081815.txt	20-Aug-2015 18:10	25M	



Load BXD genotypes

```
# Move to the data folder  
setwd("D:/CTCworkshop")
```

```
# Load the genotypes  
genotypes = read.table("BXD.geno", sep="\t", skip = 20,  
                      header = TRUE, row.names = 2,  
                      na.strings = c("U", "H"),  
                      colClasses="character")
```

```
# Dimensions of the genotype matrix  
dim(genotypes)
```



Separate the BXD genotypes from the map

```
# Get the genetic map (columns 1 to 3) and the genotype data  
map = genotypes[,1:3]
```

```
# Remove the map (columns 1 to 3) from the genotypes  
genotypes = genotypes[,-c(1:3)]
```

```
# Show a subset of the data  
map[1:10,]  
genotypes[1:10, 1:10]
```

Load the GN111 phenotypes

Load the 45.111 probes

```
GN111 = read.csv("GN111_MeanDataAnnotated_rev081815.txt",  
                skip = 33, header = TRUE, sep="\t",  
                colClasses="character")
```

Some BXD strains have two names

```
renames = c("BXD96", "BXD97", "BXD92", "BXD80", "BXD103")  
names(renames) = c("BXD48a", "BXD65a", "BXD65b", "BXD73a", "BXD73b")
```

Update the individual names of the phenotype data

```
pii = which(colnames(GN111) %in% names(renames))  
colnames(GN111)[pii] = renames[colnames(GN111)[pii]]
```



Create a phenotype subset

```
# Select two genes: Hspa4 and Hspa9a (heat shock proteins)
HSP2 = which(GN111[, "Gene.Symbol"] %in% c("Hspa4", "Hspa9a"))
GNsubset = GN111[HSP2, ]

# Dimensions of the subset
dim(GNsubset)

# Store the probe annotation
annotation = GNsubset[, c("Gene.Symbol", "Description")]
```

Match & Non-missing Genotypes

```
# Subset phenotypes and genotypes
```

```
PinG = which(colnames(GNsubset) %in% colnames(genotypes))
```

```
Gnsubset = GNsubset[, PinG]
```

```
genotypes = genotypes[, colnames(GNsubset)]
```

```
# Check the genotypes for markers no missing genotype data
```

```
allData = which(apply(apply(genotypes,1,is.na),2,sum) == 0)
```

```
# Keep markers with complete data
```

```
genotypes = genotypes[allData,]
```



Change genotype encoding

```
# Update the map, since some markers were removed
```

```
map <- map[rownames(genotypes),]
```

```
# Transpose the matrices (individuals need to be in the rows)
```

```
genotypes <- t(genotypes)
```

```
GNsubset <- t(GNsubset)
```

```
# Change encoding of the genotypes to numeric 1 and 2
```

```
genotypes[genotypes == "B"] <- 1
```

```
genotypes[genotypes == "D"] <- 2
```



Visually inspect the data

```
# Print to the R console
```

```
GNsubset[1:10, ]
```

```
annotation
```

```
genotypes[1:10, 1:5]
```

```
> GNsubset[1:10, ]
```

```
      477      478      2834      2835      15580      19502      24883      32039      38112  
BXD1 "9.378" "8.19"  "10.264" "11.006" "8.274" "9.757"  "5.041" "9.395" "5.131"  
BXD2 "9.671" "8.03"  "10.512" "11.024" "6.498" "9.811"  "4.833" "9.422" "5.953"  
BXD5 "9.558" "9.192" "10.24"  "10.996" "7.927" "9.834"  "7.712" "9.218" "5.736"  
BXD6 "9.541" "8.88"  "10.196" "11.052" "7.816" "9.906"  "5.331" "9.578" "6.363"  
BXD8 "9.742" "8.79"  "10.416" "11.156" "8.232" "9.918"  "5.984" "9.541" "5.934"
```

- * R messed up, phenotypes values are characters ?!
- * Genotypes also transformed into character ☹️



Fix R being 'smart'

Characters to numerical values

```
numgeno = apply(genotypes, 2, as.numeric)
```

```
numpheno = apply(GNsubset, 2, as.numeric)
```

Re-add the row names to the matrices

```
rownames(numgeno) = rownames(numpheno) = rownames(genotypes)
```

Check again !

```
numpheno[1:10, ]
```

```
numgeno[1:10, 1:5]
```

Data pre-processing finished

- * Data pre-processing is important
 - * Genotypes and phenotypes should fit together
 - * Same number of individuals in both
 - * Same order of individuals in both
- * Genotype data should (preferably) be non-missing
 - * The package can deal with missing genotypes, but it's better to not have missing data, when possible

Scan for CTL

```
# Scan for CTL
```

```
res_wrong = CTLscan(genotypes, GNsubset)
```

```
res_correct = CTLscan(numgeno, numpheno)
```

```
# Print out the significant CTLs
```

```
CTLsignificant(res_wrong)
```

```
CTLsignificant(res_correct)
```



Scan for CTL

```
# Scan for CTL
```

```
res_wrong = CTLscan(genotypes, GNsubset)  
res_correct = CTLscan(numgeno, numpheno)
```

```
# Print out the significant CTLs
```

```
CTLsignificant(res_wrong)  
CTLsignificant(res_correct)
```

```
> CTLsignificant(res_wrong)  
Found 8 significant CTLs  
  trait      marker trait  LOD  
1 19502  rs30804641 38112 1.97  
2 19502  rs31929413 38112 1.97  
3 19502  rs219816352 38112 1.97  
4 19502  rs50310397 38112 1.97  
5 38112  rs30804641 19502 1.97  
6 38112  rs31929413 19502 1.97  
7 38112  rs219816352 19502 1.97  
8 38112  rs50310397 19502 1.97  
> CTLsignificant(res_correct)  
Found 0 significant CTLs
```

- * Always make sure your phenotypes (and genotypes) are **numeric values !!!**



Scan them all

```
# Take the BXDs
```

```
GN111BXD = GN111[,PinG]
```

```
# Individuals into the rows
```

```
GN111BXD = t(GN111BXD)
```

```
# Convert to numeric
```

```
GN111BXD = apply(GN111BXD, 2, as.numeric)
```

```
# Try to scan all
```

```
allres <- CTLscan(numgeno, GN111BXD)
```



Limits of R

- * Scanning all phenotypes will fail because of memory constraints
- * We can solve this in two ways:
 1. Use many R instances (e.g. on a cluster) and scan a couple of phenotypes per instance
 2. Use the command line version
This writes output to the HDD, and not into the RAM

Using a cluster

- * To show this would go to far, however in general:
 1. Create an Rscript (submit1.R) that takes a cmd line parameter that tells the script which phenotype to scan
 2. Create a PBS submission script that submits submit1.R
 3. Create an R script that submits N jobs to the cluster
- * When using a cluster always turn OFF multiple testing correction ! (parameter adjust=FALSE)

Contact me (Danny.Arends@gmail.com) if you want to run CTL mapping on a cluster



Building the command line version

- * In windows, add the Rtools C compiler to the path

- * E.g.: C:\Rtools\mingw_64\bin

- * Checkout CTLmapping from Github

```
git clone https://github.com/DannyArends/CTLmapping.git
```

- * Go into the C folder inside the CTL folder and build the executable:

```
make R_HOME="C:\Program Files\R\R-3.3.2"
```



Building the command line version

- * Compile the code and a mapctl.exe is created

```
D:\Ddrive\Github\CTLmapping>cd C
D:\Ddrive\Github\CTLmapping\C>make R_HOME="C:\Program Files\R\R-3.3.2"
rm -rf *o *.so *.a *exe summary.txt
rm -rf ../Rctl/src/*o
gcc -I "C:\Program Files\R\R-3.3.2/include/" -std=gnu99 -O3 -pedantic -w -Wall -Wextra -c -I. -I../Rctl/src
gcc -I "C:\Program Files\R\R-3.3.2/include/" -std=gnu99 -O3 -pedantic -w -Wall -Wextra -c -I. -I../Rctl/src
gcc -I "C:\Program Files\R\R-3.3.2/include/" -std=gnu99 -O3 -pedantic -w -Wall -Wextra -c -I. -I../Rctl/src
gcc -I "C:\Program Files\R\R-3.3.2/include/" -std=gnu99 -O3 -pedantic -w -Wall -Wextra -c -I. -I../Rctl/src
gcc -I "C:\Program Files\R\R-3.3.2/include/" -std=gnu99 -O3 -pedantic -w -Wall -Wextra -c -I. -I../Rctl/src
gcc -I "C:\Program Files\R\R-3.3.2/include/" -std=gnu99 -O3 -pedantic -w -Wall -Wextra -c -I. -I../Rctl/src
gcc -I "C:\Program Files\R\R-3.3.2/include/" -std=gnu99 -O3 -pedantic -w -Wall -Wextra -c -I. -I../Rctl/src
gcc -I "C:\Program Files\R\R-3.3.2/include/" -std=gnu99 -O3 -pedantic -w -Wall -Wextra -c -I. -I../Rctl/src
gcc -I "C:\Program Files\R\R-3.3.2/include/" -std=gnu99 -O3 -pedantic -w -Wall -Wextra -c -I. -I../Rctl/src
gcc -I "C:\Program Files\R\R-3.3.2/include/" -std=gnu99 -O3 -pedantic -w -Wall -Wextra -c -I. -I../Rctl/src
gcc -lm -L "C:\Program Files\R\R-3.3.2/bin/x64/" -lR -lRblas -O3 main.o ../Rctl/src/vector.o ../Rctl/src/matr
tl
```

Building the command line version

- * Executing the executable with `-help` shows the help

```
D:\Ddrive\Github\CTLmapping\C>mapctl.exe --help
Correlated Trait Locus (CTL) mapping
(c) 2012-2020 GBIC - RUG & HU-Berlin, written by Danny Arends
Number of command line arguments passed: 1
mapctl.exe: unknown option -- -
Usage:
mapctl -g<genotype_file> -p<phenotype_file>

-p<FILE>   Input file with phenotype data (Default: phenotypes.csv)
-g<FILE>   Input file with genotype data (Default: genotypes.csv)
-o<FILE>   Name of the output file (Default: summary.txt)
-t<N>     Significance threshold (0..1) (Default: 0.01)
-f         Save all the results to file
-d         When specified permutation are performed
-n<N>     # of permutations (Default: 100)
-h         Shows this help
```



Example input files

- * Examples of input files for the command line version

<https://github.com/DannyArends/CTLmapping/tree/master/D/test/data>

- * The command line version does NOT adjusted for multiple testing

- * Summary contains

- * LOD

- * Correlations

```
1 Trait>Marker→Trait>LOD>Cor_0>Cor_1
2 0>74→3>2.35→-0.509→-0.104
3 0>53→4>2.36→0.005>0.453
4 0>54→4>2.23→0.006>0.437
5 0>55→4>2.32→0.011>0.447
6 0>67→4>2.22→-0.009→0.409
7 0>94→4>2.03→0.415→-0.009
8 0>100→4>6.44→0.663→-0.027
9 0>11→5>2.51→0.041>0.475
10 0>12→5>2.85→0.030>0.494
11 0>13→5>2.10→0.089>0.473
12 0>14→5>2.89→0.049>0.511
13 0>16→5>2.27→0.056>0.467
14 0>17→5>3.03→0.033>0.517
15 0>18→5>2.78→0.044>0.505
16 0>19→5>3.08→0.030>0.525
```



Session overview

- * CTL mapping on BXD
 - * Data is 'easily' converted to the CTL package format
 - * Scanning can be done using the R version
 - * When there are a huge amount of phenotypes
 - * Use a cluster
 - * Use the standalone version
- * C and D command line executables are available