Seattle Summer Institute 2012

15: Systems Genetics for Experimental Crosses Tutorial Notes

Brian S. Yandell, UW-Madison

Elias Chaibub Neto, Sage Bionetworks

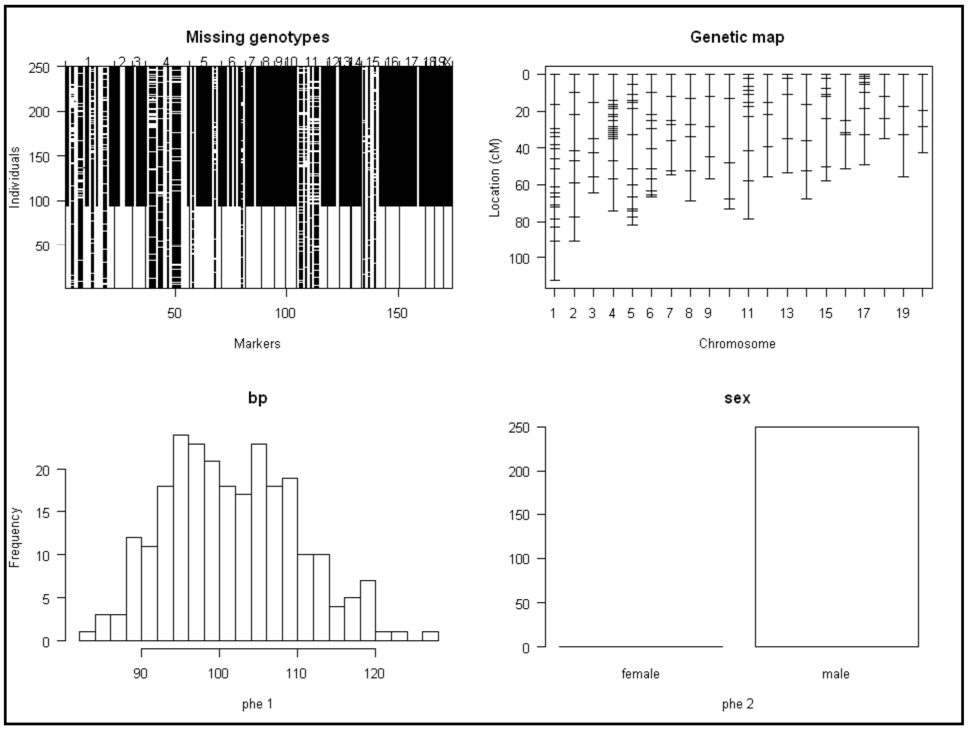
www.stat.wisc.edu/~yandell/statgen/sisg

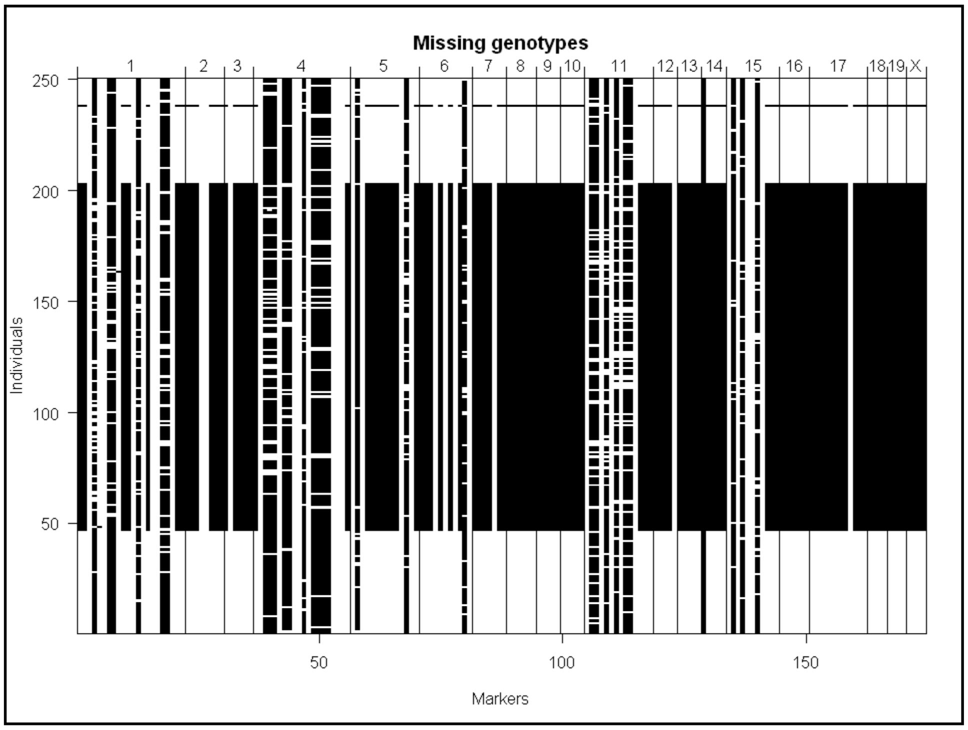
R/qtl & R/qtlbim Tutorials

- R statistical graphics & language system
- R/qtl tutorial
  - R/qtl web site: www.rqtl.org
  - Tutorial: www.rqtl.org/tutorials/rqtltour.pdf
  - R code: www.stat.wisc.edu/~yandell/qtlbim/rqtltour.R
  - url.show("http://www.stat.wisc.edu/~yandell/qtlbim/rqtltour.R")
- R/qtlbim tutorial
  - R/qtlbim web site: www.qtlbim.org
  - Tutorial and R code:
    - www.stat.wisc.edu/~yandell/qtlbim/rqtlbimtour.pdf
    - www.stat.wisc.edu/~yandell/qtlbim/rqtlbimtour.R

R/qtl tutorial (www.rqtl.org)

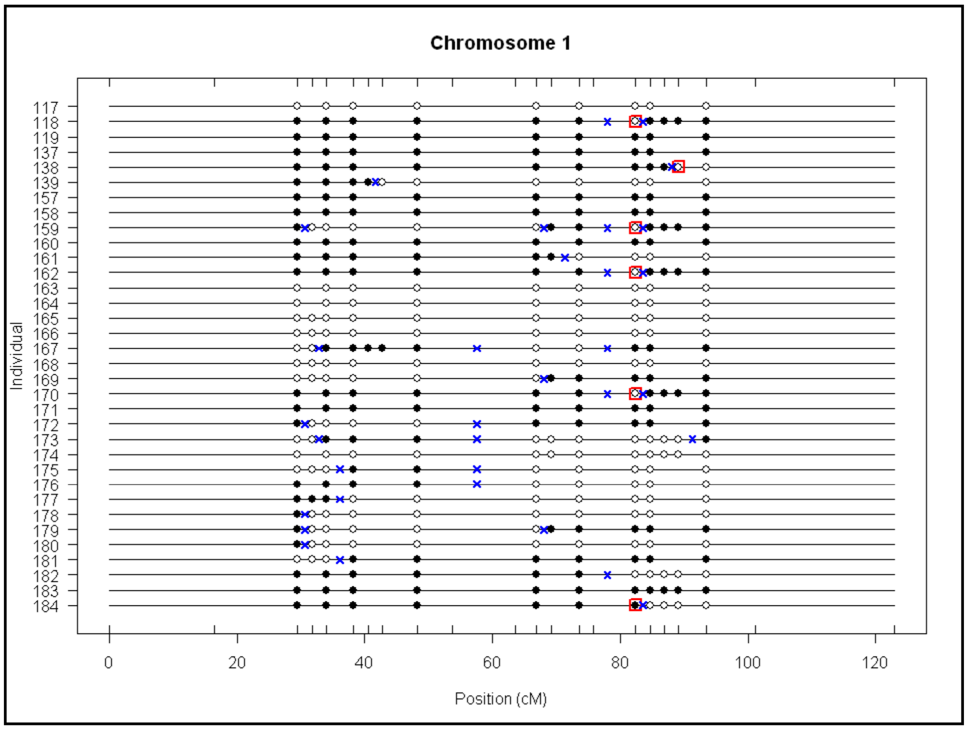
```
> library(qtl)
> data(hyper)
> summary(hyper)
    Backcross

    No. individuals:    250

    No. phenotypes:     2
    Percent phenotyped: 100 100

    No. chromosomes:    20
        Autosomes:      1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
        X chr:          X

    Total markers:      174
    No. markers:        22 8 6 20 14 11 7 6 5 5 14 5 5 5 11 6 12 4 4 4
    Percent genotyped:  47.7
    Genotypes (%):      AA:50.2  AB:49.8
> plot(hyper)
> plot.missing(hyper, reorder = TRUE)
```





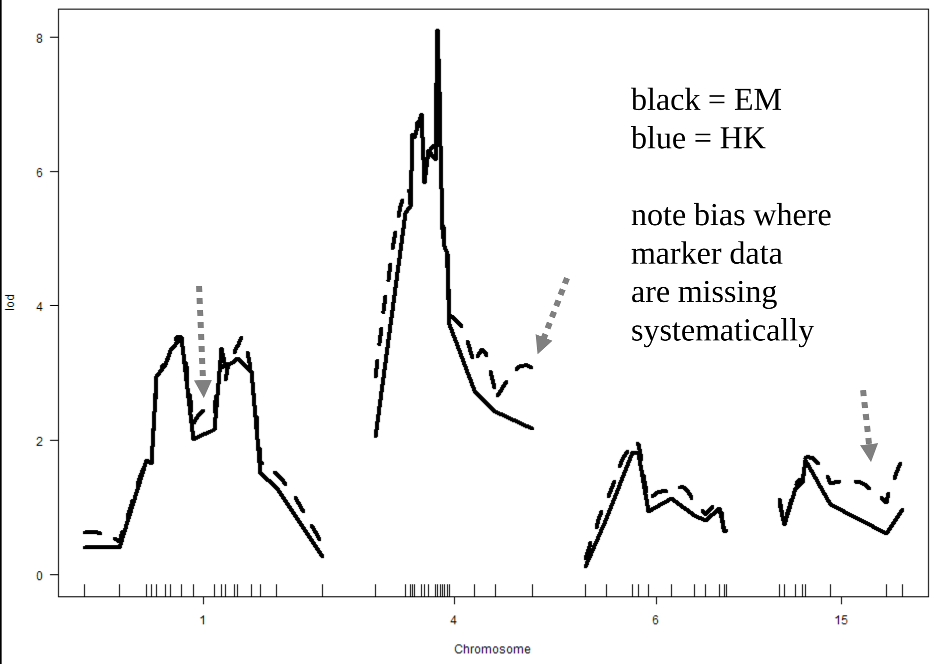
R/qtl: find genotyping errors

```
> hyper <- calc.errorlod(hyper, error.prob=0.01)
> top.errorlod(hyper)

   chr  id    marker errorlod
1    1 118   D1Mit14 8.372794
2    1 162   D1Mit14 8.372794
3    1 170   D1Mit14 8.372794
4    1 159   D1Mit14 8.350341
5    1  73   D1Mit14 6.165395
6    1  65   D1Mit14 6.165395
7    1  88   D1Mit14 6.165395
8    1 184   D1Mit14 6.151606
9    1 241   D1Mit14 6.151606
...
16   1 215  D1Mit267 5.822192
17   1 108  D1Mit267 5.822192
18   1 138  D1Mit267 5.822192
19   1 226  D1Mit267 5.822192
20   1 199  D1Mit267 5.819250
21   1  84  D1Mit267 5.808400

> plot.geno(hyper, chr=1, ind=c(117:119,137:139,157:184))
```



R/qtl: 1 QTL interval mapping

```
> hyper <- calc.genoprob(hyper, step=1,
  error.prob=0.01)
> out.em <- scanone(hyper)
> out.hk <- scanone(hyper, method="hk")
> summary(out.em, threshold=3)
         chr  pos  lod
c1.loc45   1 48.3 3.52
D4Mit164   4 29.5 8.02

> summary(out.hk, threshold=3)
         chr  pos  lod
c1.loc45   1 48.3 3.55
D4Mit164   4 29.5 8.09

> plot(out.em, chr = c(1,4,6,15))
> plot(out.hk, chr = c(1,4,6,15), add = TRUE, lty = 2)
```

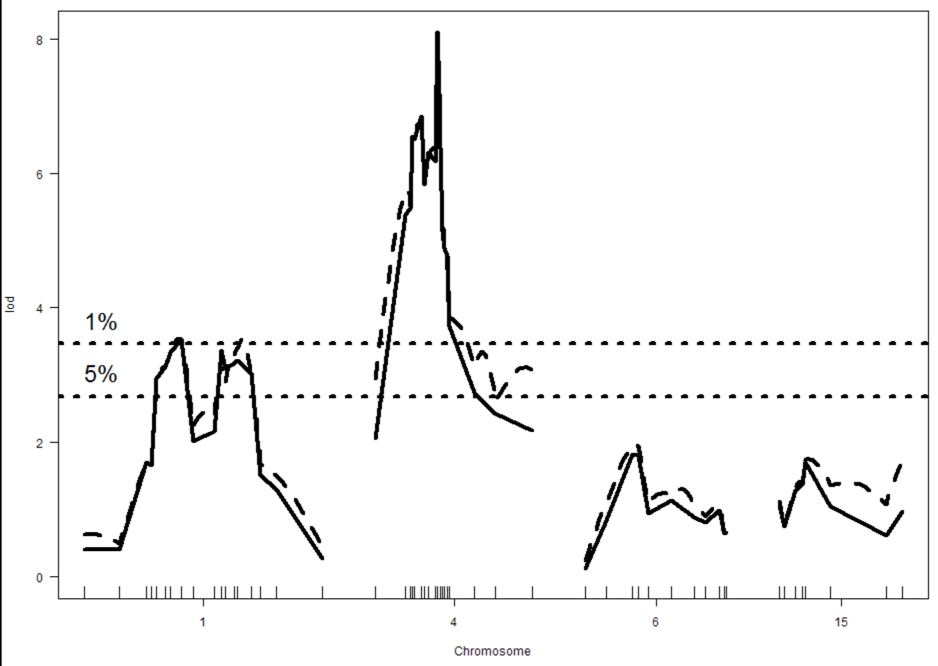


black = EM
blue = HK

note bias where
marker data
are missing
systematically

R/qtl: permutation threshold

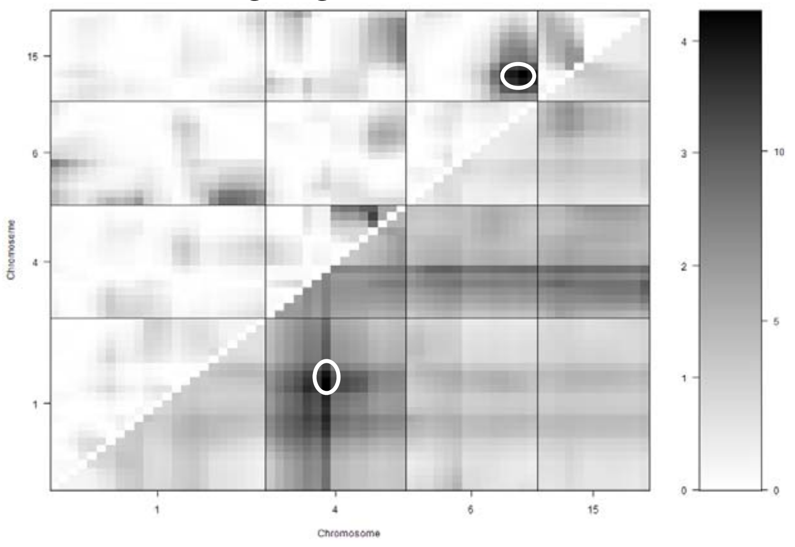
```
> operm.hk <- scanone(hyper, method="hk",
  n.perm=1000)
Doing permutation in batch mode ...
> summary(operm.hk, alpha=c(0.01,0.05))
LOD thresholds (1000 permutations)
    lod
1% 3.79
5% 2.78

> summary(out.hk, perms=operm.hk, alpha=0.05,
  pvalues=TRUE)
  chr  pos  lod  pval
1   1 48.3 3.55 0.015
2   4 29.5 8.09 0.000
```



R/qtl: 2 QTL scan

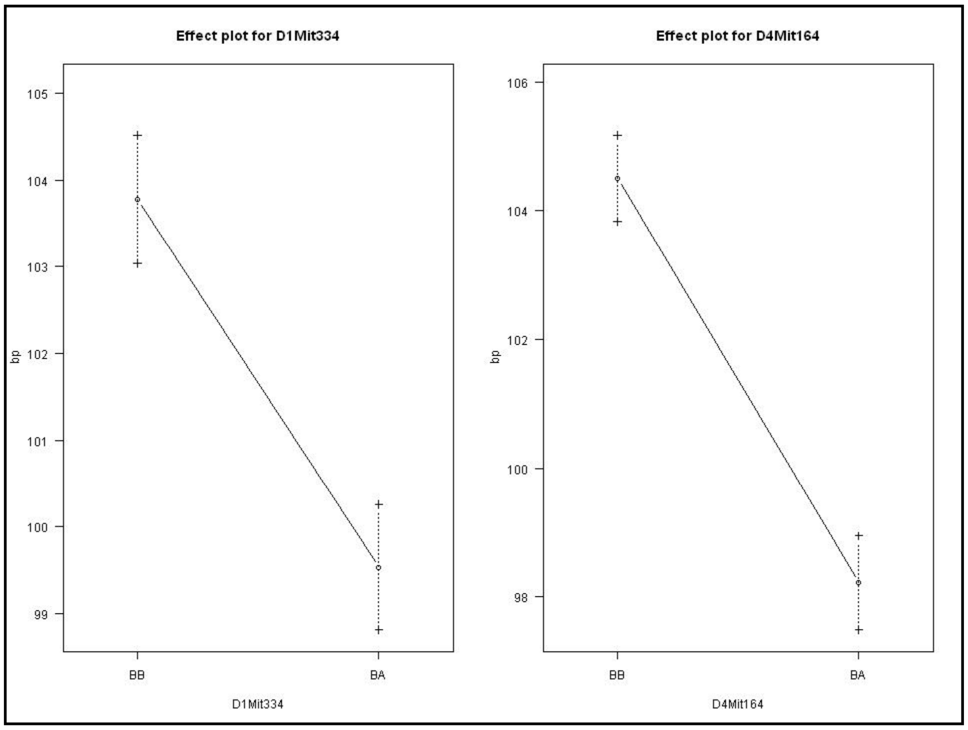
```
> hyper <- calc.genoprob(hyper, step=5, error.prob=0.01)
>
> out2.hk <- scantwo(hyper, method="hk")
 --Running scanone
 --Running scantwo
 (1,1)
 (1,2)
...
 (19,19)
 (19,X)
 (X,X)
> summary(out2.hk, thresholds=c(6.0, 4.7, 4.4, 4.7, 2.6))

        pos1f pos2f lod.full lod.fv1 lod.int     pos1a pos2a lod.add lod.av1
c1 :c4   68.3  30.0    14.13    6.51   0.225      68.3  30.0   13.90   6.288
c2 :c19  47.7   0.0     6.71    5.01   3.458      52.7   0.0    3.25   1.552
c3 :c3   37.2  42.2     6.10    5.08   0.226      37.2  42.2    5.87   4.853
c6 :c15  60.0  20.5     7.17    5.22   3.237      25.0  20.5    3.93   1.984
c9 :c18  67.0  37.2     6.31    4.79   4.083      67.0  12.2    2.23   0.708
c12:c19   1.1  40.0     6.48    4.79   4.090       1.1   0.0    2.39   0.697

> plot(out2.hk, chr=c(1,4,6,15))
```

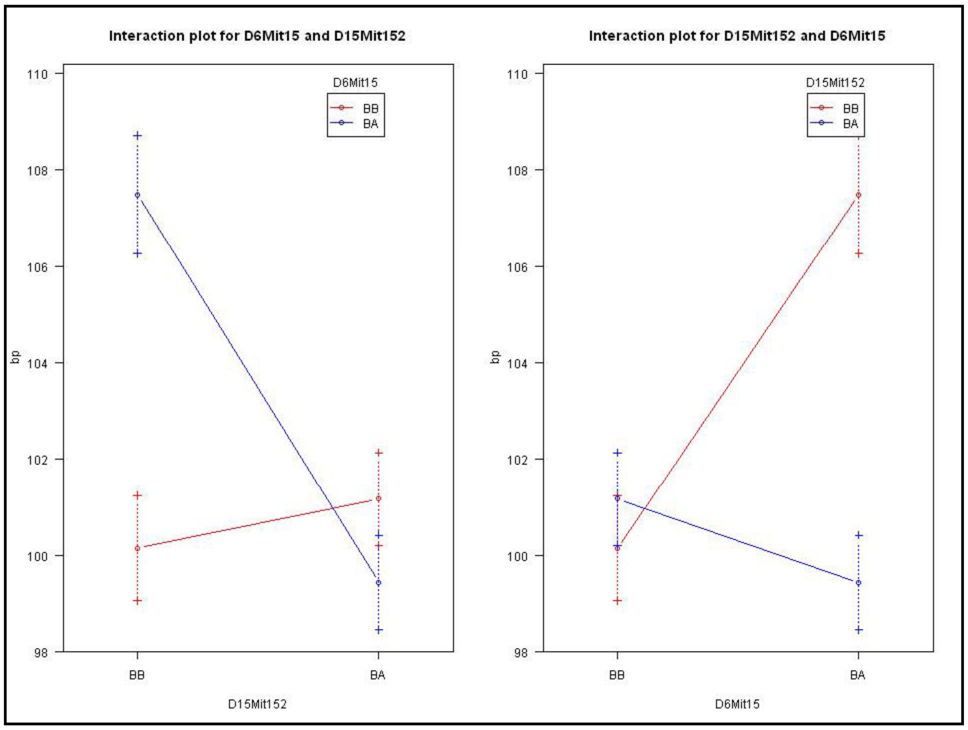
upper triangle/left scale: epistasis LOD
lower triangle/right scale: 2-QTL LOD

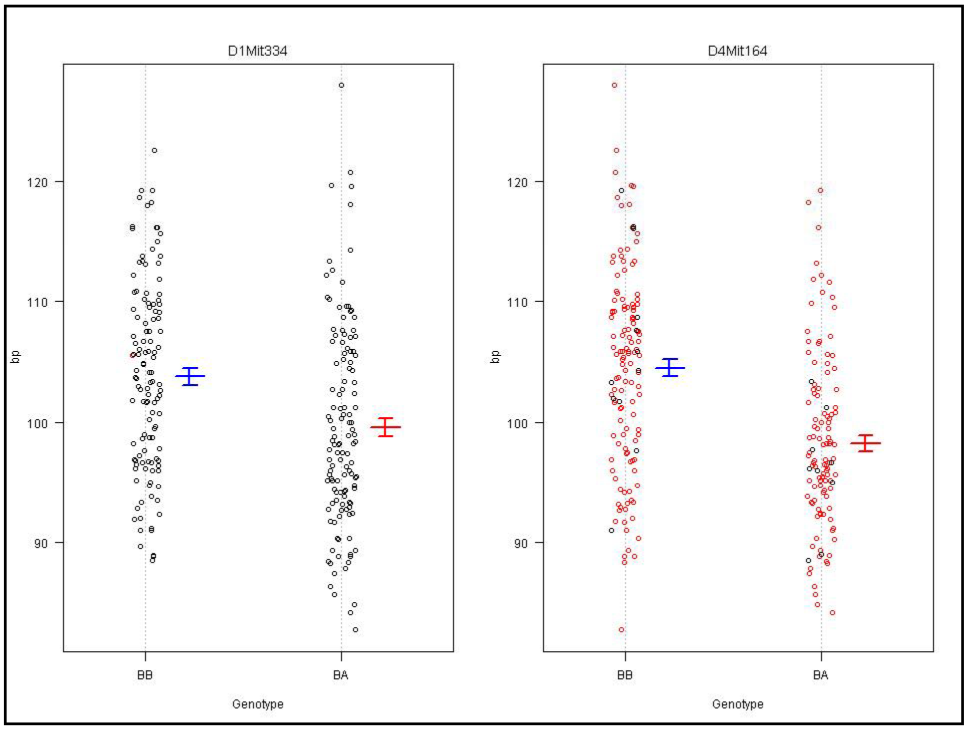


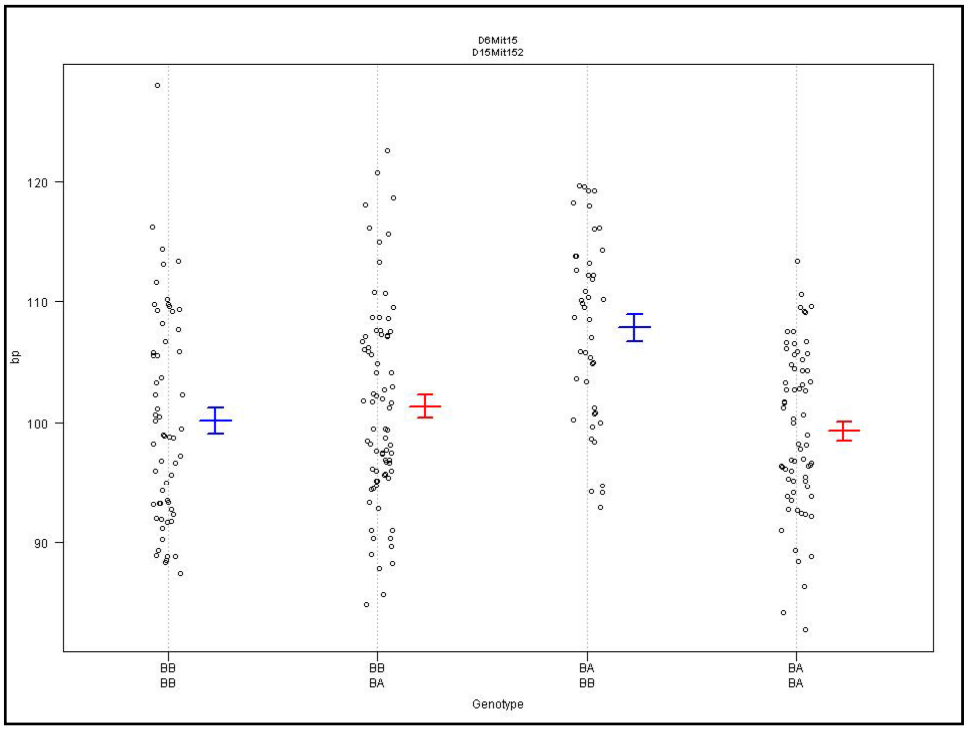
Effect & Interaction Plots

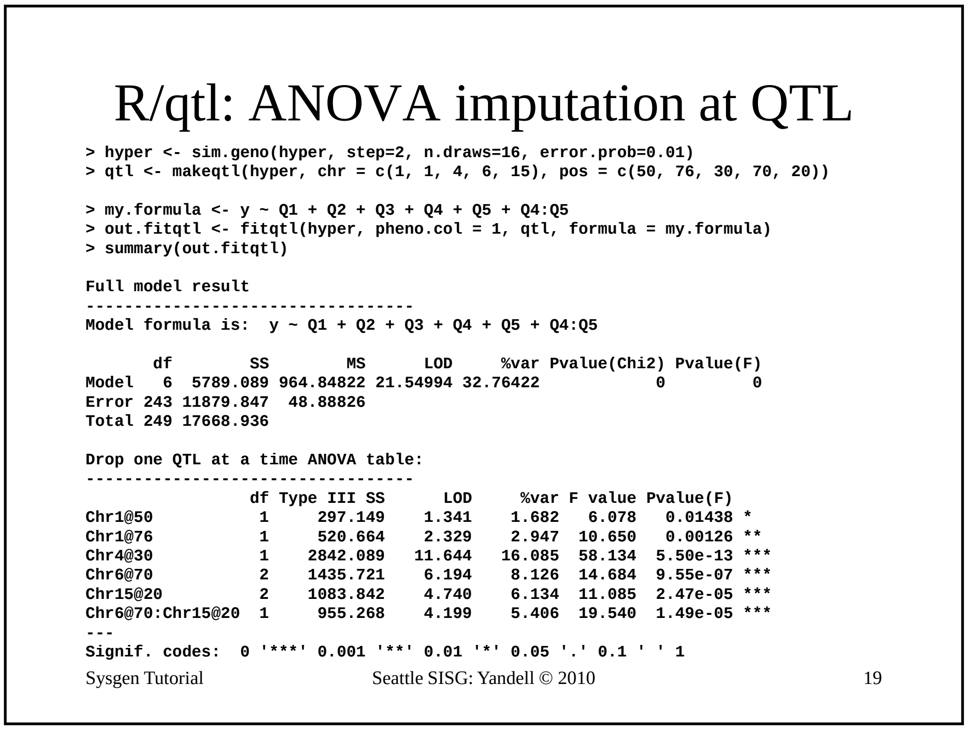
```
## Effect plots and interaction plot.
hyper <- sim.geno(hyper, step=2, n.draws=16, error.prob=0.01)
effectplot(hyper, pheno.col = 1, mname1 = "D1Mit334")
effectplot(hyper, pheno.col = 1, mname1 = "D4Mit164")
markers <- find.marker(hyper, chr = c(6,15), pos = c(70,20))
effectplot(hyper, pheno.col = 1,
   mname1 = markers[1], mname2 = markers[2])
effectplot(hyper, pheno.col = 1,
   mname1 = markers[2], mname2 = markers[1])

## Strip plot of data (phenotype by genotype).
plot.pxg(hyper, "D1Mit334")
plot.pxg(hyper, "D4Mit164")
plot.pxg(hyper, markers)
```


Effect plot for D1Mit334
Effect plot for D4Mit164
bp
105
104
103
102
101
100
99
106
98
BB
BA
D1Mit334
D4Mit164


Interaction plot for D6Mit15 and D15Mit152
Interaction plot for D15Mit152 and D6Mit15
D6Mit15
D15Mit152
BB
BA
bp
110
108
106
104
102
100
98
D15Mit152
D6Mit15

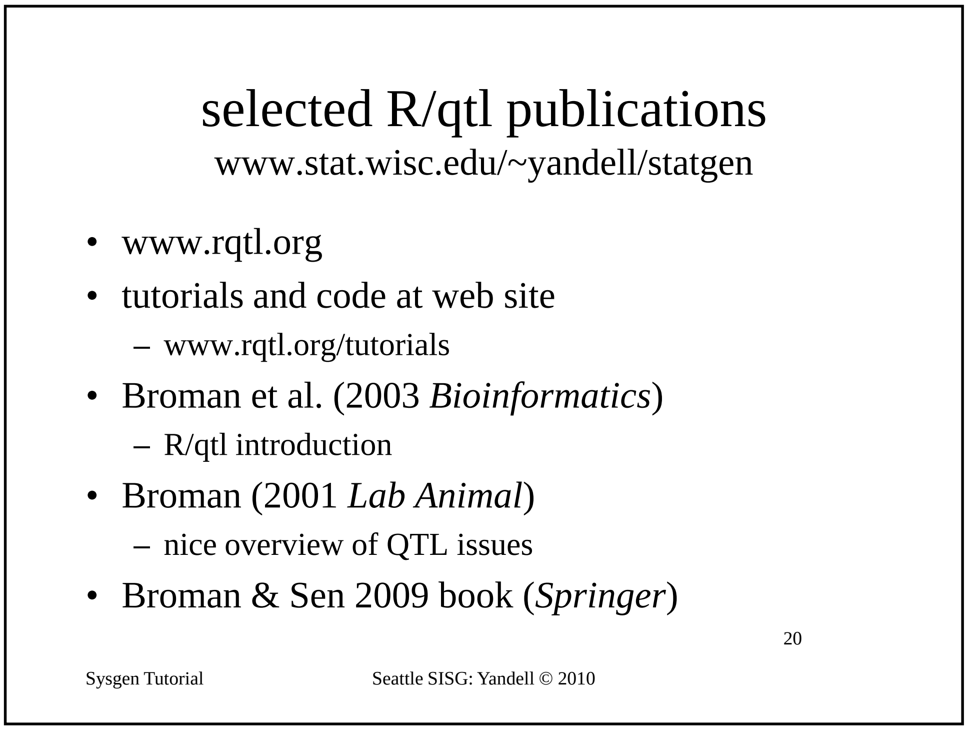

D1Mit334
D4Mit164
bp
120
110
100
90
BB
BA
Genotype


D6Mit15
D15Mit152
bp
120
110
100
90
BB BB
BB BA
BA BB
BA BA
Genotype



R/qtl: ANOVA imputation at QTL

```
> hyper <- sim.geno(hyper, step=2, n.draws=16, error.prob=0.01)
> qtl <- makeqtl(hyper, chr = c(1, 1, 4, 6, 15), pos = c(50, 76, 30, 70, 20))

> my.formula <- y ~ Q1 + Q2 + Q3 + Q4 + Q5 + Q4:Q5
> out.fitqtl <- fitqtl(hyper, pheno.col = 1, qtl, formula = my.formula)
> summary(out.fitqtl)

Full model result
----------------------------------
Model formula is:  y ~ Q1 + Q2 + Q3 + Q4 + Q5 + Q4:Q5

       df        SS        MS      LOD     %var Pvalue(Chi2) Pvalue(F)
Model   6  5789.089 964.84822 21.54994 32.76422            0         0
Error 243 11879.847  48.88826
Total 249 17668.936

Drop one QTL at a time ANOVA table:
----------------------------------
                 df Type III SS      LOD     %var F value Pvalue(F)
Chr1@50           1     297.149    1.341    1.682   6.078   0.01438 *
Chr1@76           1     520.664    2.329    2.947  10.650   0.00126 **
Chr4@30           1    2842.089   11.644   16.085  58.134  5.50e-13 ***
Chr6@70           2    1435.721    6.194    8.126  14.684  9.55e-07 ***
Chr15@20          2    1083.842    4.740    6.134  11.085  2.47e-05 ***
Chr6@70:Chr15@20  1     955.268    4.199    5.406  19.540  1.49e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```



selected R/qtl publications

www.stat.wisc.edu/~yandell/statgen

- www.rqtl.org
- tutorials and code at web site
  - www.rqtl.org/tutorials
- Broman et al. (2003 *Bioinformatics*)
  - R/qtl introduction
- Broman (2001 *Lab Animal*)
  - nice overview of QTL issues
- Broman & Sen 2009 book (*Springer*)

R/qtlbim (www.qtlbim.org)

- cross-compatible with R/qtl
- model selection for genetic architecture
  - – epistasis, fixed & random covariates, GxE
  - – samples multiple genetic architectures
  - – examines summaries over nested models
- extensive graphics

```
> url.show("http://www.stat.wisc.edu/~yandell/qtlbim/rqtlbimtour.R")
```

R/qtlbim: tutorial

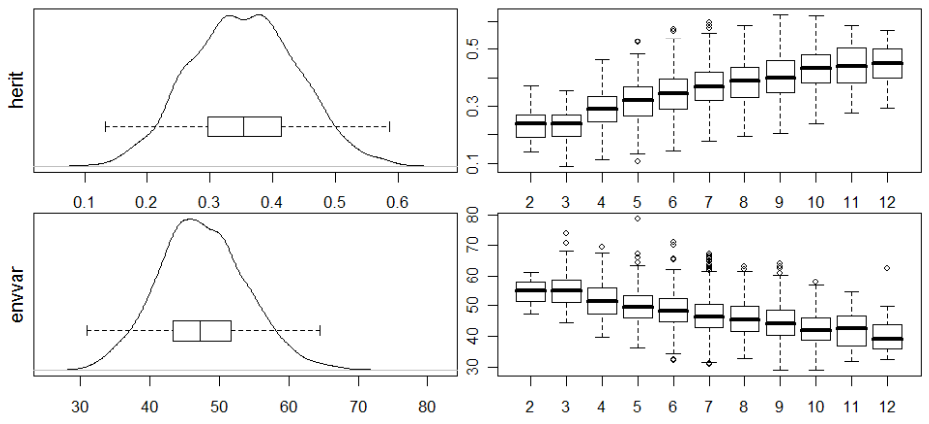
(www.stat.wisc.edu/~yandell/qtlbim)

```
> data(hyper)
## Drop X chromosome (for now).
> hyper <- subset(hyper, chr=1:19)
> hyper <- qb.genoprob(hyper, step=2)
## This is the time-consuming step:
> qbHyper <- qb.mcmc(hyper, pheno.col = 1)
## Here we get stored samples.
> data(qbHyper)
> summary(qbHyper)
```

R/qtlbim: initial summaries

```
> summary(qbHyper)

Bayesian model selection QTL mapping object qbHyper on cross object hyper
had 3000 iterations recorded at each 40 steps with 1200 burn-in steps.

Diagnostic summaries:
          nqtl   mean envvar varadd  varaa    var
Min.     2.000  97.42  28.07  5.112  0.000  5.112
1st Qu.  5.000 101.00  44.33 17.010  1.639 20.180
Median   7.000 101.30  48.57 20.060  4.580 25.160
Mean     6.543 101.30  48.80 20.310  5.321 25.630
3rd Qu.  8.000 101.70  53.11 23.480  7.862 30.370
Max.    13.000 103.90  74.03 51.730 34.940 65.220

Percentages for number of QTL detected:
 2  3  4  5  6  7  8  9 10 11 12 13
 2  3  9 14 21 19 17 10  4  1  0  0

Percentages for number of epistatic pairs detected:
pairs
 1  2  3  4  5  6
29 31 23 11  5  1

Percentages for common epistatic pairs:
 6.15  4.15   4.6   1.7 15.15   1.4   1.6   4.9  1.15  1.17   1.5  5.11   1.2  7.15   1.1
   63    18    10     6     6     5     4     4     3     3     3     2     2     2     2

> plot(qb.diag(qbHyper, items = c("herit", "envvar")))
```

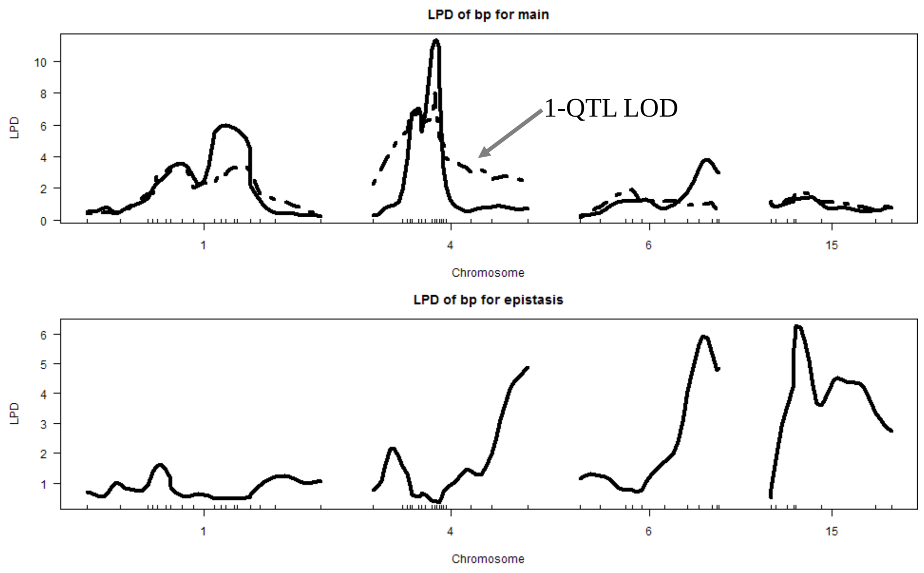
diagnostic summaries



R/qtlbim: 1-D (*not* 1-QTL!) scan

```
> one <- qb.scanone(qbHyper, chr = c(1,4,6,15), type =
   "LPD")
> summary(one)

LPD of bp for main,epistasis,sum

    n.qtl  pos m.pos e.pos  main epistasis   sum
c1  1.331 64.5  64.5  67.8  6.10     0.442  6.27
c4  1.377 29.5  29.5  29.5 11.49     0.375 11.61
c6  0.838 59.0  59.0  59.0  3.99     6.265  9.60
c15 0.961 17.5  17.5  17.5  1.30     6.325  7.28

> plot(one, scan = "main")
> plot(out.em, chr=c(1,4,6,15), add = TRUE, lty = 2)
> plot(one, scan = "epistasis")
```

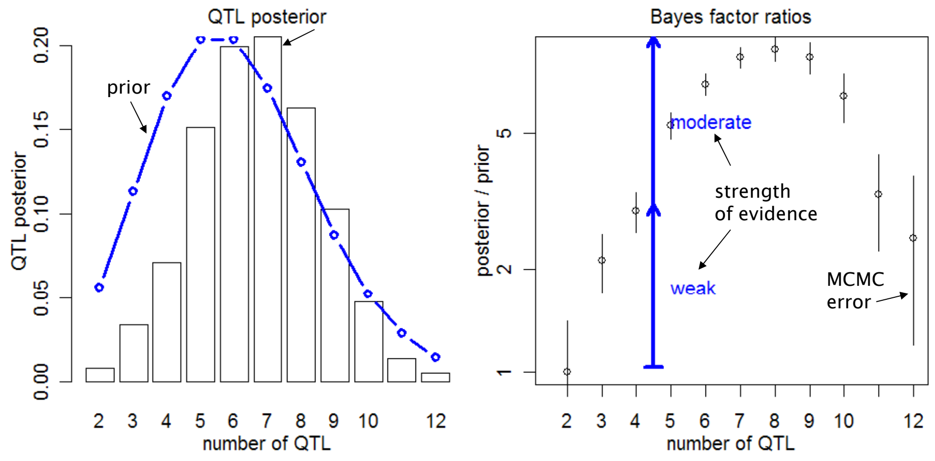
1-QTL LOD vs. marginal LPD



most probable patterns

```
> summary(qb.BayesFactor(qbHyper, item = "pattern"))

                  nqtl posterior    prior    bf  bfse
1,4,6,15,6:15        5   0.03400 2.71e-05 24.30 2.360
1,4,6,6,15,6:15      6   0.00467 5.22e-06 17.40 4.630
1,1,4,6,15,6:15      6   0.00600 9.05e-06 12.80 3.020
1,1,4,5,6,15,6:15    7   0.00267 4.11e-06 12.60 4.450
1,4,6,15,15,6:15     6   0.00300 4.96e-06 11.70 3.910
1,4,4,6,15,6:15      6   0.00300 5.81e-06 10.00 3.330
1,2,4,6,15,6:15      6   0.00767 1.54e-05  9.66 2.010
1,4,5,6,15,6:15      6   0.00500 1.28e-05  7.56 1.950
1,2,4,5,6,15,6:15    7   0.00267 6.98e-06  7.41 2.620
1,4                  2   0.01430 1.51e-04  1.84 0.279
1,1,2,4              4   0.00300 3.66e-05  1.59 0.529
1,2,4                3   0.00733 1.03e-04  1.38 0.294
1,1,4                3   0.00400 6.05e-05  1.28 0.370
1,4,19               3   0.00300 5.82e-05  1.00 0.333

> plot(qb.BayesFactor(qbHyper, item = "nqtl"))
```

hyper: number of QTL posterior, prior, Bayes factors

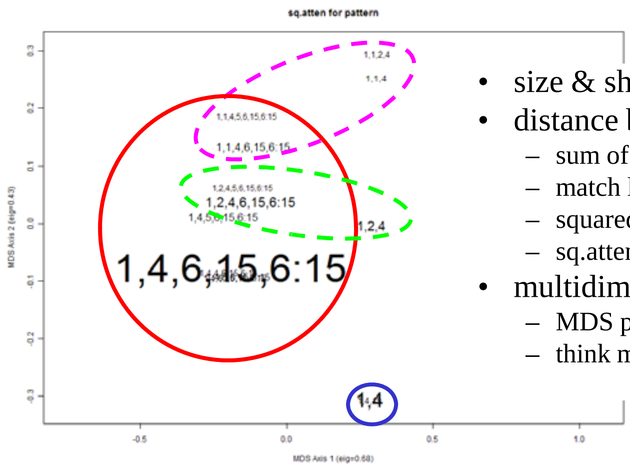


what is best estimate of QTL?

- find most probable pattern
  - 1,4,6,15,6:15 has posterior of 3.4%
- estimate locus across all nested patterns
  - Exact pattern seen ~100/3000 samples
  - Nested pattern seen ~2000/3000 samples
- estimate 95% confidence interval using quantiles

```
> best <- qb.best(qbHyper)
> summary(best)$best

    chrom locus locus.LCL locus.UCL     n.qtl
247     1  69.9  24.44875   95.7985 0.8026667
245     4  29.5  14.20000   74.3000 0.8800000
248     6  59.0  13.83333   66.7000 0.7096667
246    15  19.5  13.10000   55.7000 0.8450000

> plot(best)
```

what patterns are “near” the best?

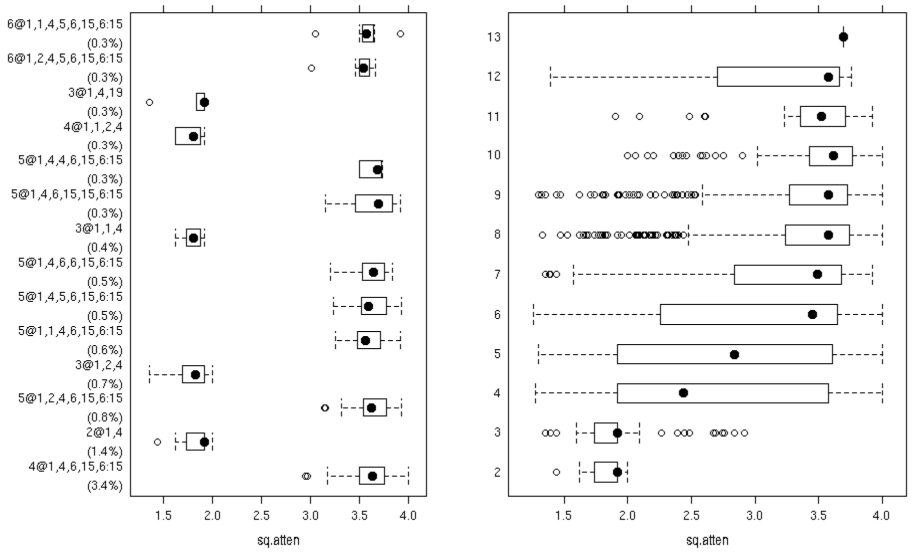


- size & shade ~ posterior
- distance between patterns
  - sum of squared attenuation
  - match loci between patterns
  - squared attenuation = $(1-2r)^2$
  - sq.atten in scale of LOD & LPD
- multidimensional scaling
  - MDS projects distance onto 2-D
  - think mileage between cities

how close are other patterns?

```
> target <- qb.best(qbHyper)$model[[1]]
> summary(qb.close(qbHyper, target))

 score by sample number of qtl
    Min. 1st Qu. Median  Mean 3rd Qu.  Max.
2  1.437   1.735  1.919 1.834   1.919 2.000
3  1.351   1.735  1.916 1.900   1.919 2.916
4  1.270   1.916  2.437 2.648   3.574 4.000
5  1.295   1.919  2.835 2.798   3.611 4.000
6  1.257   2.254  3.451 3.029   3.648 4.000
...
13 3.694   3.694  3.694 3.694   3.694 3.694

 score by sample chromosome pattern
                    Percent  Min. 1st Qu. Median  Mean 3rd Qu.  Max.
4@1,4,6,15,6:15         3.4 2.946   3.500  3.630 3.613   3.758 4.000
2@1,4                   1.4 1.437   1.735  1.919 1.832   1.919 2.000
5@1,2,4,6,15,6:15       0.8 3.137   3.536  3.622 3.611   3.777 3.923
3@1,2,4                 0.7 1.351   1.700  1.821 1.808   1.919 2.000
5@1,1,4,6,15,6:15       0.6 3.257   3.484  3.563 3.575   3.698 3.916
5@1,4,5,6,15,6:15       0.5 3.237   3.515  3.595 3.622   3.777 3.923
5@1,4,6,6,15,6:15       0.5 3.203   3.541  3.646 3.631   3.757 3.835
...

> plot(close)
> plot(close, category = "nqtl")
```

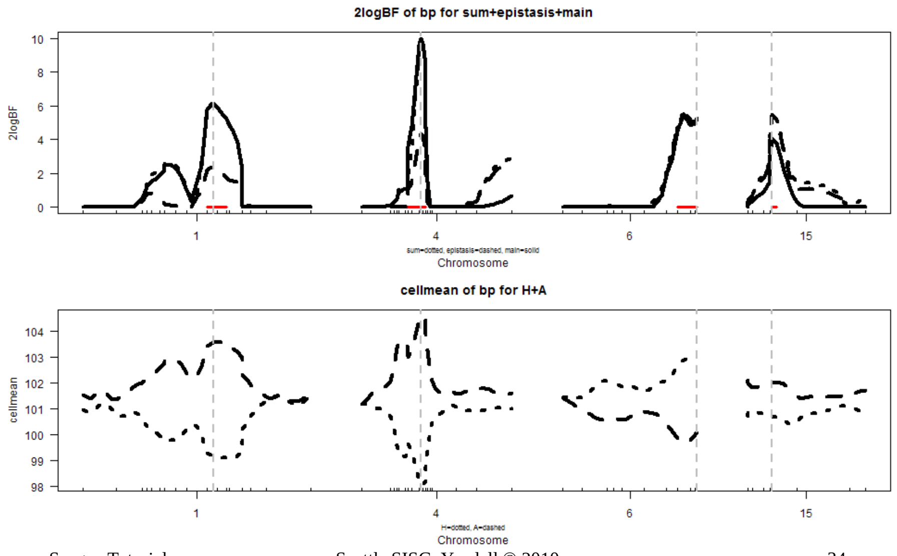
how close are other patterns?



R/qtlbim: automated QTL selection

```
> hpd <- qb.hpdone(qbHyper, profile = "2logBF")
> summary(hpd)

   chr n.qtl  pos lo.50% hi.50% 2logBF       A       H
1    1 0.829 64.5   64.5   72.1  6.692 103.611  99.090
4    4 3.228 29.5   25.1   31.7 11.169 104.584  98.020
6    6 1.033 59.0   56.8   66.7  6.054  99.637 102.965
15  15 0.159 17.5   17.5   17.5  5.837 101.972 100.702

> plot(hpd)
```

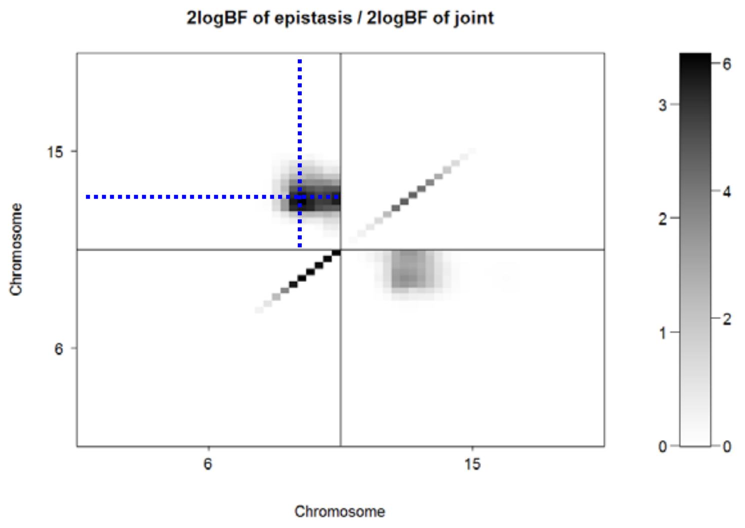
2log(BF) scan with 50% HPD region



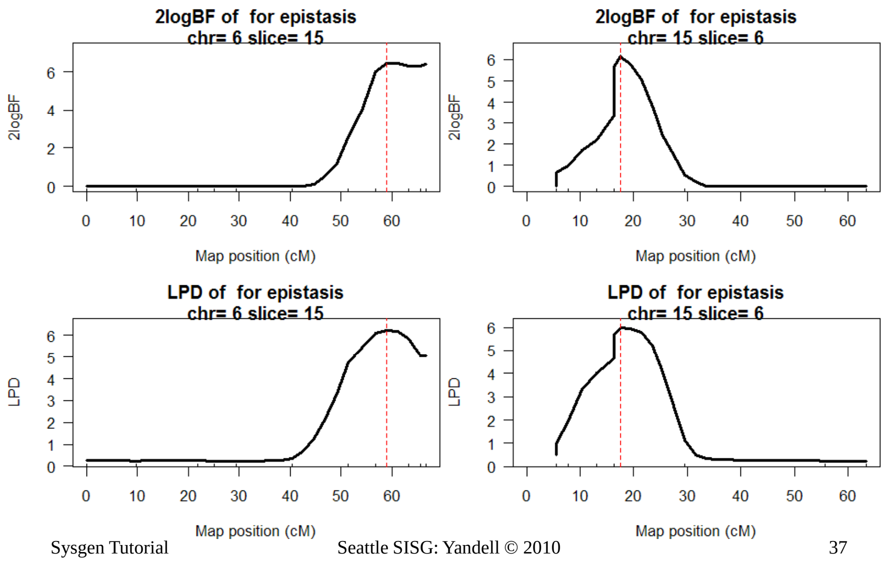
R/qtlbim: 2-D (*not* 2-QTL) scans

```
> two <- qb.scantwo(qbHyper, chr = c(6,15),
  type = "2logBF")
> plot(two)

> plot(two, chr = 6, slice = 15)
> plot(two, chr = 15, slice = 6)

> two.lpd <- qb.scantwo(qbHyper, chr = c(6,15),
  type = "LPD")
> plot(two.lpd, chr = 6, slice = 15)
> plot(two.lpd, chr = 15, slice = 6)
```

2-D plot of 2logBF: chr 6 & 15

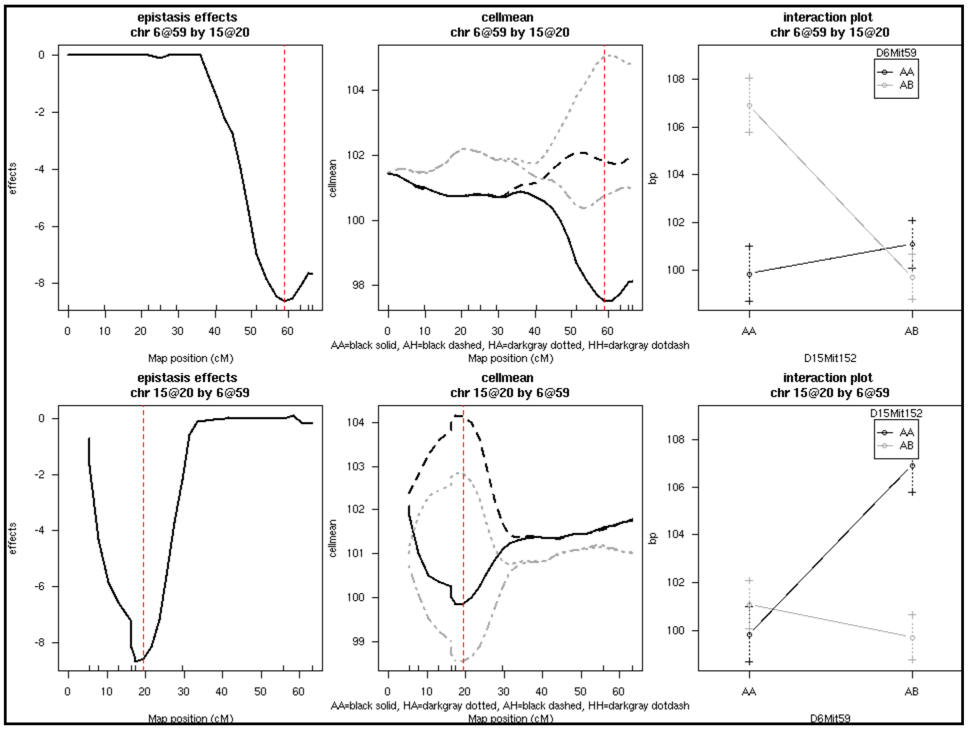


1-D Slices of 2-D scans: chr 6 & 15



R/qtlbim: slice of epistasis

```
> slice <- qb.slicetwo(qbHyper, c(6,15), c(59,19.5))
> summary(slice)

2logBF of bp for epistasis

    n.qtl  pos m.pos e.pos epistasis slice
c6  0.838 59.0  59.0  66.7      15.8  18.1
c15 0.961 17.5  17.5  17.5      15.5  60.6

cellmean of bp for AA,HA,AH,HH

    n.qtl  pos m.pos   AA  HA  AH    HH slice
c6  0.838 59.0  59.0 97.4 105 102 100.8  18.1
c15 0.961 17.5  17.5 99.8 103 104  98.5  60.6

estimate of bp for epistasis

    n.qtl  pos m.pos e.pos epistasis slice
c6  0.838 59.0  59.0  66.7     -7.86  18.1
c15 0.961 17.5  17.5  17.5     -8.72  60.6

> plot(slice, figs = c("effects", "cellmean", "effectplot"))
```



selected publications

www.stat.wisc.edu/~yandell/statgen

- www.qtlbim.org
- vignettes in R/qtlbim package
- Yandell, Bradbury (2007) *Plant Map* book chapter
  - overview/comparison of QTL methods
- Yandell et al. (2007 *Bioinformatics*)
  - R/qtlbim introduction
- Yi et al. (2005 *Genetics*, 2007 *Genetics*)
  - methodology of R/qtlbim

The R/qtlhot package

Elias Chaibub Neto and Brian S Yandell

SISG 2012

July 9, 2012

Simulate a “null” cross

Start by simulating a “null backcross” composed of 1,000 phenotypes, 204 genetic markers equally spaced across 4 chr, and 100 ind. The **latent.eff** parameter controls the amount of correlation among the phenotypes.

```
> library(qtlhot)
> ncross1 <- sim.null.cross(chr.len = rep(100, 4),
+                           n.mar = 51,
+                           n.ind = 100,
+                           type = "bc",
+                           n.pheno = 1000,
+                           latent.eff = 3,
+                           res.var = 1,
+                           init.seed = 123457)
```

Include hotspots into null cross

The function **include.hotspots** takes the "null cross" as an input and includes 3 hotspots of size **hsize** at position **hpos** of chromosome **hchr** into it.

```
> cross1 <- include.hotspots(cross = ncross1,
+                            hchr = c(2, 3, 4),
+                            hpos = c(25, 75, 50),
+                            hsize = c(100, 50, 20),
+                            Q.eff = 2,
+                            latent.eff = 3,
+                            lod.range.1 = c(2.5, 2.5),
+                            lod.range.2 = c(5, 8),
+                            lod.range.3 = c(10, 15),
+                            res.var = 1,
+                            nT = 1000,
+                            init.seed = 12345)
```

Check correlation among phenotypes

By choosing **latent.eff** we generate highly correlated phenotype data.

```
> nphe1 <- as.matrix(cross1$pheno)
> ncor1 <- cor(nphe1)
> ncor1 <- ncor1[lower.tri(ncor1)]
> summary(ncor1)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.4145  0.8517  0.8929  0.8649  0.9063  0.9691
```

Single trait QTL mapping permutation threshold

Obtain permutation thresholds for the sequence **alphas** of GWER levels.

```
> set.seed(123)
> pt <- scanone(ncross1, method = "hk", n.perm = 1000)
> alphas <- seq(0.01, 0.10, by=0.01)
> spt <- summary(pt, alphas)
> spt
LOD thresholds (1000 permutations)
     lod
1%  3.11
2%  2.89
3%  2.68
4%  2.57
5%  2.44
6%  2.34
7%  2.26
8%  2.20
9%  2.15
10% 2.11
> lod.thrs <- as.vector(spt)
```

QTL mapping and LOD profile processing

Perform QTL mapping analysis using H-K regression, and processing of the LOD profiles by setting to zero LOD values outside the 1.5 LOD support interval around the peak at each chromosome (as well as LOD values below the single trait mapping threshold, **thr**).

```
> scan1 <- scanone(cross1, pheno.col = 1:1000, method = "hk")
> scandrop1 <- set.to.zero.beyond.drop.int(chr = scan1[,1],
+                      scanmat = as.matrix(scan1[,-c(1,2)]),
+                      thr = min(lod.thrs),
+                      drop = 1.5)
```

By setting to zero the LOD scores outside the LOD support interval we can considerably decrease the spread of the hotspot.

Hotspot architecture at varying thresholds

For each genomic position, we count the number of traits with LOD $\geq$ **lod.thrs**.

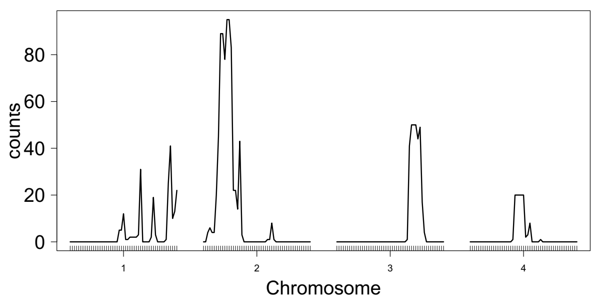
The counts1 object is a matrix with 204 rows (genetic markers) and 10 columns (thresholds).

```
> counts1 <- t(count.thr(scandrop1, lod.thrs, droptwo = FALSE))
> counts1[52:56,]
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
D2M1    0    0    0    0    0    0    0    0    0     0
D2M2    0    0    0    0    0    0    0    0    0     0
D2M3    0    1    2    3    4    5    6    8   13    15
D2M4    2    2    3    5    6   14   17   21   24    27
D2M5    0    2    3    3    4    6    8   11   13    14
```

The first column gives the counts for threshold of 3.11. The last one shows the counts for threshold 2.11. Note how the counts increase as the QTL mapping thresholds decrease.

Hotspot architecture for LOD thr 2.44 ($\alpha = 0.05$)

```
> out1 <- data.frame(scan1[, 1:2], counts1)
> class(out1) <- c("scanone", "data.frame")
> par(mar=c(4.1,4.1,0.1,0.1))
> plot(out1, lodcolumn = 5, ylab = "counts", cex.lab = 1.5,
+      cex.axis = 1.5)
```



Note the spurious hotspots on chr 1.

Q-method

The **WW.perm** function implements the *Q*-method's permutation scheme.

```
> set.seed(12345)
> Q.1 <- WW.perm(scanmat = scandrop1,
+                lod.thrs = lod.thrs,
+                n.perm = 100,
+                verbose = FALSE)
```

The output is a matrix with 100 rows (permutations), and 10 columns (thresholds). Each entry *ij*, represents the maximum number of significant linkages across the entire genome detected at permutation *i*, using the LOD threshold *j*.

Q-method

The **WW.summary** function computes the hotspot size permutation thresholds.

```
> Q.1.thr <- WW.summary(Q.1, alphas)
> Q.1.thr
                  0.01  0.02  0.03  0.04 0.05  0.06  0.07  0.08  0.09  0.1
3.10508056313925 11.00 10.02 10.00 10.00   10 10.00 10.00 10.00 10.00 10.0
2.89135162173146 12.00 12.00 11.03 11.00   11 11.00 11.00 11.00 11.00 11.0
2.67690269000741 14.01 13.02 13.00 13.00   13 13.00 13.00 13.00 13.00 13.0
2.5743266994317  16.01 16.00 16.00 15.04   15 14.06 14.00 14.00 14.00 14.0
2.43869721183317 18.00 18.00 17.03 17.00   17 17.00 17.00 17.00 17.00 16.1
2.335067939838   21.01 21.00 20.03 20.00   20 20.00 19.07 19.00 19.00 19.0
2.2577747088154  22.02 22.00 22.00 21.04   21 21.00 21.00 20.08 20.00 20.0
2.19884780562269 23.01 23.00 22.03 22.00   22 22.00 22.00 22.00 22.00 21.1
2.15023439516803 24.02 24.00 24.00 23.04   23 23.00 23.00 23.00 22.09 22.0
2.11039422475441 26.02 26.00 25.03 25.00   25 25.00 24.07 24.00 24.00 24.0
```

Q-method

In general, we are interested in using the same error rates for the QTL mapping and hotspot analysis.

Therefore, we are usually more interested on the diagonal of **Q.1.thr**.

For the hotspots depicted in the previous figure, we adopted a GWER of 5%, and the corresponding *Q*-method's permutation threshold is 17.

According to this threshold, all hotspots are significant.

```
> diag(Q.1.thr)
 [1] 11.00 12.00 13.00 15.04 17.00 20.00 21.00 22.00 22.09 24.00
```

N- and *NL*-methods

The **NL.N.perm** function implements the *N*- and *NL*-methods' permutation schemes.

The argument **Nmax** sets the maximum hotspot size to be analyzed by the *NL*-method.

The argument **drop** controls the magnitude of the LOD support interval computation during the LOD profile processing step.

```
> set.seed(12345)
> NL.N.1 <- NL.N.perm(cross = cross1,
+                     Nmax = 300,
+                     n.perm = 100,
+                     lod.thrs = lod.thrs,
+                     drop = 1.5,
+                     verbose = TRUE)
> names(NL.N.1)
[1] "max.lod.quant" "max.N"
```

The function's output is a list with two elements: **max.lod.quant** and **max.N**.

N- and *NL*-methods

max.lod.quant stores the output of the *NL*-method's perms. It is given by a matrix with 100 rows (permutations), and 300 columns (hotspot sizes analyzed).

Entry *ij* stores the maximum genome wide $qLOD(n)$ computed at permutation i using threshold j, where $qLOD(n)$ corresponds to the nth LOD score in a sample ordered from highest to lowest.

For instance, consider the first 3 lines and 6 columns of **max.lod.quant**. At the 3rd permutation, the maximum LOD score across the genome was 3.37, the second maximum across the genome was 3.36, and so on.

```
> NL.N.1[[1]][1:3, 1:6]
            1        2        3        4        5        6
[1,] 2.115918 1.903466 1.713409 1.649016 1.600378 1.594265
[2,] 2.464650 2.162832 1.932474 1.885934 1.878833 1.839507
[3,] 3.374947 3.358949 3.198482 3.195974 3.121577 3.105578
```

N- and *NL*-methods

max.N stores the output of the *N*-method's perms. It is given by a matrix with 100 rows (permutations), and 10 columns (thresholds).

Entry *ij* stores the maximum genome wide hotspot size detected at permutation i when computed using threshold j (note the output is transposed).

```
> t(NL.N.1[[2]][1:6,])
                 [,1] [,2] [,3] [,4] [,5] [,6]
3.10508056313925    0    0    6    0   19    9
2.89135162173146    0    0   14    0   31   18
2.67690269000741    0    0   26    2   45   34
2.5743266994317     0    0   40    4   52   60
2.43869721183317    0    1   65   13   66   97
2.335067939838      0    1   83   25   81  158
2.2577747088154     0    1  106   36   90  213
2.19884780562269    0    1  131   46  101  249
2.15023439516803    0    2  162   61  116  290
2.11039422475441    1    2  186   75  127  328
```

N- and *NL*-methods

The **NL.N.summary** function computes the *N*- and *NL*-method's hotspot size permutation thresholds.

```
> NL.N.1.thrs <- NL.N.summary(NL.N.1[[1]], NL.N.1[[2]], alphas)
> NL.1.thr <- NL.N.1.thrs[[1]]
> N.1.thr <- NL.N.1.thrs[[2]]
```

N- and *NL*-methods

N.1.thr is a 10 by 10 matrix with rows indexing the QTL mapping thr and columns indexing the target GWER.

Each entry *ij* shows the hotspot size above which a hotspot is considered significant at a GWER *j* using the QTL mapping threshold *i*.

The *N*-method's threshold that controls the hotspot GWER at a 5% level when the QTL mapping was controlled at a GWER of 5% is 195.75.

```
> N.1.thr[1:3, ]
                   0.01   0.02   0.03   0.04  0.05  0.06  0.07  0.08  0.09
3.10508056313925  52.23  46.08  35.33  32.12 25.35 25.00 20.35 19.08 18.09
2.89135162173146  95.06  86.12  83.09  53.24 39.65 39.00 31.56 30.08 29.09
2.67690269000741 191.59 180.16 157.69 103.24 86.75 65.32 59.35 51.64 46.45
> diag(N.1.thr)
 [1]  52.23  86.12 157.69 138.68 195.75 191.50 228.49 250.28 272.71 286.60
```

According to the *N*-method, none of the hotspots in the previous figure is significant.

N- and *NL*-methods

The **NL.1.thr** object is a matrix with 300 rows (spurious hotspot sizes analyzed), and 10 columns (target GWER).

Each entry *ij* represents the LOD threshold at which a hotspot of size greater or equal than *i* is significant at a GWER less or equal to *j*.

```
> round(NL.1.thr[1:3,], 4)
    0.01   0.02   0.03   0.04   0.05   0.06   0.07   0.08   0.09    0.1
1 4.8767 4.7365 4.4521 4.3385 4.1959 4.1198 4.0367 3.9752 3.9376 3.7978
2 4.4265 4.3883 4.3569 3.8245 3.7798 3.7610 3.7364 3.6578 3.6093 3.5616
3 4.3150 4.1852 4.1284 3.8023 3.7285 3.6702 3.6643 3.6173 3.5022 3.4818
```

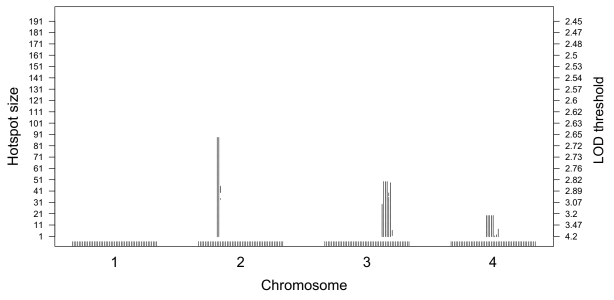
N- and *NL*-methods

Hotspot significance profile.

```
> N.1 <- round(N.1.thr[5, 5])
> par(mar=c(4.1,4.1,0.1,4.1))
> sliding.bar.plot(scan = data.frame(scan1[, 1:2], scandrop1),
+                  lod.thr = NL.1.thr[1:N.1, 5],
+                  size.thr = 1:N.1,
+                  gap = 50,
+                  y.axes = seq(1, N.1, by = 10))
```

N- and *NL*-methods

For each genomic location this figure shows the hotspot sizes at which the hotspot was significant, that is, at which the hotspot locus had more traits than the hotspot size threshold on the left mapping to it with a LOD score higher than the threshold on the right than expected by chance.



The R/qtlcmst package

Elias Chaibub Neto and Brian S Yandell

SISG 2012

July 10, 2012

Simulate data

We first use the **SimCrossCausal** function to simulate a `cross` object with 3 phenotypes, y_1, y_2 and y_3, where y_1 has a causal effect on both y_2 and y_3.

```
> set.seed(987654321)
> Cross <- SimCrossCausal(n.ind = 100,
+                          len = rep(100, 3),
+                          n.mar = 101,
+                          beta = rep(0.5, 2),
+                          add.eff = 1,
+                          dom.eff = 0,
+                          sig2.1 = 0.4,
+                          sig2.2 = 0.1,
+                          eq.spacing = FALSE,
+                          cross.type = "bc",
+                          normalize = TRUE)
```

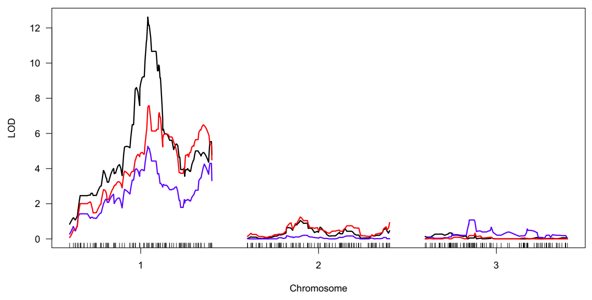
QTL mapping

Compute the genotype conditional probabilities setting the maximum distance between positions at which genotype probabilities were calculated to 1cM.

```
> Cross <- calc.genoprob(Cross, step = 1)
```

Perform QTL mapping using Haley-Knott regression.

```
> Scan <- scanone(Cross, pheno.col = 1:3, method = "hk")
> plot(Scan, lodcolumn = 1:3, ylab = "LOD")
```



Black, blue and red curves represent phenos y_1, y_2 and y_3, respectively.

QTL mapping

Summarize the results for the 3 phenotypes.

```
> summary(Scan[, c(1, 2, 3)], thr = 3)
        chr pos   y1
c1.loc55   1  55 12.6
> summary(Scan[, c(1, 2, 4)], thr = 3)
        chr pos   y2
c1.loc55   1  55 5.27
> summary(Scan[, c(1, 2, 5)], thr = 3)
      chr  pos   y3
D1M50   1 55.5 7.58
```

y_1 and y_2 map to the same QTL at position 55 cM on chr 1, y_3 maps to a distinct position.

Which QTL should we use as causal anchor?

QTL mapping

Our approach is to compute the joint LOD profile of both phenos and use the QTL detected by this joint approach as the causal anchor.

```
> commqtls <- GetCommonQtls(Cross,
+                           pheno1 = "y1",
+                           pheno2 = "y3",
+                           thr = 3,
+                           peak.dist = 5,
+                           addcov1 = NULL,
+                           addcov2 = NULL,
+                           intcov1 = NULL,
+                           intcov2 = NULL)
> commqtls
         Q Q.chr Q.pos
1 c1.loc55     1    55
```

CMST tests

Fit the CMST tests.

```
> nms <- names(Cross$pheno)
> out1 <- CMSTtests(Cross,
+                   pheno1 = nms[1],
+                   pheno2 = nms[2],
+                   Q.chr = 1,
+                   Q.pos = 55,
+                   addcov1 = NULL,
+                   addcov2 = NULL,
+                   intcov1 = NULL,
+                   intcov2 = NULL,
+                   cross.type = "bc",
+                   method = "all",
+                   penalty = "both")
```

CMST tests - output

```
> out1[1:6]
$pheno1
[1] "y1"

$pheno2
[1] "y2"

$n.ind
[1] 100

$loglik
[1] -123.5318 -140.4604 -141.5803 -123.4834

$model.dim
[1] 6 6 6 7

$R2
[1] 0.4407170 0.2153583
```

CMST tests - output

Covariance matrix of the log-likelihood scores.

```
> out1[7]
$S.hat
            [,1]        [,2]         [,3]         [,4]         [,5]
[1,]  0.26221327 -0.01323094  0.010924311 -0.275444212 -0.251288963
[2,] -0.01323094  0.36275299  0.012080993  0.375983930  0.025311930
[3,]  0.01092431  0.01208099  0.001115354  0.001156681 -0.009808958
[4,] -0.27544421  0.37598393  0.001156681  0.651428142  0.276600893
[5,] -0.25128896  0.02531193 -0.009808958  0.276600893  0.241480006
[6,]  0.02415525 -0.35067200 -0.010965639 -0.374827248 -0.035120888
            [,6]
[1,]  0.02415525
[2,] -0.35067200
[3,] -0.01096564
[4,] -0.37482725
[5,] -0.03512089
[6,]  0.33970636
```

CMST tests - output

```
> out1[8:12]
$BICs
[1] 274.6946 308.5518 310.7917 279.2030

$Z.bic
     [,1]     [,2]      [,3]      [,4]
[1,]   NA 3.305926 2.9966507  6.749745
[2,]   NA       NA 0.1387598 -2.986200
[3,]   NA       NA        NA -2.709873
[4,]   NA       NA        NA        NA

$pvals.p.BIC
[1] 0.001364817 0.999526684 0.998635183 1.000000000

$pvals.np.BIC
[1] 6.289575e-06 9.999977e-01 9.999999e-01 1.000000e+00

$pvals.j.BIC
[1] 0.003779558 0.999946885 0.999669186 1.000000000
```

CMST tests - output

```
> out1[13:17]
$AICs
[1] 259.0636 292.9208 295.1606 260.9668

$Z.aic
     [,1]     [,2]      [,3]      [,4]
[1,]   NA 3.305926 2.9966507  2.849429
[2,]   NA       NA 0.1387598 -3.251273
[3,]   NA       NA        NA -2.933361
[4,]   NA       NA        NA        NA

$pvals.p.AIC
[1] 0.002189889 0.999526684 0.998635183 0.997810111

$pvals.np.AIC
[1] 6.289575e-06 9.999977e-01 1.000000e+00 9.999977e-01

$pvals.j.AIC
[1] 0.005993868 0.999946885 0.999669186 1.000000000
```

CMST tests

Fit one phenotype against a list of phenotypes.

```
> out2 <- CMSTtestsList(Cross,
+                       pheno1 = nms[1],
+                       phenos = nms[-1],
+                       Q.chr = 1,
+                       Q.pos = 55,
+                       addcov1 = NULL,
+                       addcov2 = NULL,
+                       intcov1 = NULL,
+                       intcov2 = NULL,
+                       cross.type = "bc",
+                       method = "par",
+                       penalty = "bic")
```

CMST tests

```
> out2
$R2s
      R2.Y1 ~ Q R2.Y2 ~ Q
y1_y2  0.440717 0.2153583
y1_y3  0.440717 0.2914979

$BIC.stats
         BIC.1    BIC.2    BIC.3    BIC.4     z.12     z.13     z.14
y1_y2 274.6946 308.5518 310.7917 279.2030 3.305926 2.996651 6.749745
y1_y3 270.4445 294.0943 325.3707 274.6665 2.339472 4.207872 3.446223
           z.23      z.24      z.34
y1_y2 0.1387598 -2.986200 -2.709873
y1_y3 1.9587743 -2.126754 -4.070649

$pvals.p.BIC
           pval.1    pval.2    pval.3    pval.4
y1_y2 0.001364817 0.9995267 0.9986352 1.0000000
y1_y3 0.009655499 0.9903445 0.9999871 0.9997158
```

The R/qtlnet package

Elias Chaibub Neto and Brian S Yandell

SISG 2012

July 8, 2012

Simulate data

We simulate data from a F_2 cross with 500 ind, and 5 chr of len 100 cM, containing 11 equally spaced markers per chr. We simulated one QTL per pheno. The QTLs, Q_t, $t = 1, 2, 3, 4, 5$, were placed at the middle marker on chr t. We set additive and dominance QTL effects to 1 and 0, respectively.

```
> library(qtlnet)
> set.seed(12345)
> Map <- sim.map(len = rep(100, 5), n.mar = 11, eq.spacing = TRUE,
+                include.x = FALSE)
> Cross <- sim.cross(map = Map, n.ind = 500, type = "f2")
> crosses <- vector(mode = "list", length = 5)
> add.effects <- c(1, 1, 1, 1, 1)
> for (i in 1:5) {
+   map <- sim.map(len = rep(100, i), n.mar = 11, eq.spacing = TRUE,
+                        include.x = FALSE)
+   crosses[[i]] <- sim.cross(map = map, n.ind = 500, type = "f2",
+                             model = c(i, 50, add.effects[i], 0))
+   Cross$geno[[i]] <- crosses[[i]]$geno[[i]]
+ }
```

Simulate data

The pheno data was simulated according to the network below, using regr equations with regr coeffs set to 1.

```
> beta <- 1
> Cross$pheno[, 1] <- crosses[[1]]$pheno
> Cross$pheno[, 2] <- crosses[[2]]$pheno + beta * Cross$pheno[, 1]
> Cross$pheno[, 3] <- crosses[[3]]$pheno + beta * Cross$pheno[, 2]
> Cross$pheno[, 4] <- crosses[[4]]$pheno + beta * Cross$pheno[, 2]
> Cross$pheno[, 5] <- crosses[[5]]$pheno + beta * Cross$pheno[, 3] +
+                     beta * Cross$pheno[,4]
> names(Cross$pheno) <- paste("y", 1:5, sep = "")
```

$$\begin{tikzcd}
 & & & Q_3 \arrow[d] & & \\
 & & & Y_3 \arrow[dr] & & \\
Q_1 \arrow[r] & Y_1 \arrow[r] & Y_2 \arrow[ur] \arrow[dr] & & Y_5 & Q_5 \arrow[l] \\
 & & Q_2 \arrow[u] & Y_4 \arrow[ur] & & \\
 & & & Q_4 \arrow[u] & &
\end{tikzcd}$$

Permutation test threshold

We determine the QTL mapping LOD threshold via permutation test.

```
> Cross <- calc.genoprob(Cross, step = 1)
> set.seed(12345)
> perm.test <- scanone(Cross, n.perm = 1000, method = "hk")
Doing permutation in batch mode ...
> summary(perm.test)
LOD thresholds (1000 permutations)
     lod
5%  3.04
10% 2.70
```

We adopt a LOD threshold of 3.04, that aims to control GWER $< 5\%$.

QDG routines

We perform QTL mapping with Haley-Knott regression for all 5 phenotypes.

```
> Scan <- scanone(Cross, pheno.col = 1:5, method = "hk")
```

Next we determine the QTLs for each phenotype, and create a list with objects of class **qtl** that is needed as impute for the **qdg** function.

```
> Cross <- sim.geno(Cross, n.draws = 1)
> marker.nms <- allqtls <- vector(mode = "list", length = 5)
> names(marker.nms) <- names(allqtls) <- paste("y", 1:5, sep = "")
> for (i in 1:5) {
+   aux <- summary(Scan[, c(1, 2, i + 2)], thr = 3.04)
+   marker.nms[[i]] <- find.marker(Cross, chr = aux[, 1], pos = aux[, 2])
+   allqtls[[i]] <- makeqtl(Cross, chr = aux[, 1], pos = aux[, 2])
+ }
```

QDG routines

Fit the QDG algorithm.

```
> out1 <- qdg(cross = Cross,
+             phenotype.names = paste("y", 1:5, sep = ""),
+             marker.names = marker.nms,
+             QTL = allqtls,
+             alpha = 0.005,
+             n.qdg.random.starts = 10,
+             addcov = NULL,
+             intcov = NULL,
+             skel.method = "pcskel")
>
> out1$UDG
  node1 node2 edge
1    y1    y2    1
3    y2    y3    1
4    y2    y4    1
6    y3    y5    1
8    y4    y5    1
```

QDG routines

```
> out1$DG
  node1 direction node2 lod score
1    y1     ---->    y2 24.135325
2    y2     ---->    y3 23.990280
3    y2     ---->    y4 32.013798
4    y3     ---->    y5  3.119176
5    y4     ---->    y5  9.726617
>
>out1$Solutions
$solutions
$solutions[[1]]
  node1 direction node2      lod
1    y1     ---->    y2 24.13533
2    y2     ---->    y3 61.02425
3    y2     ---->    y4 69.14849
4    y3     ---->    y5 54.17467
5    y4     ---->    y5 69.25563

$loglikelihood
[1] -3595.164
```

QDG routines

Plot the QDGs

```
> gr1 <- graph.qdg(out1, include.qtl = FALSE)
> plot(gr1)
> gr2 <- graph.qdg(out1, include.qtl = TRUE)
> plot(gr2)
```

(cannot export eps from R. pdf has no margins)

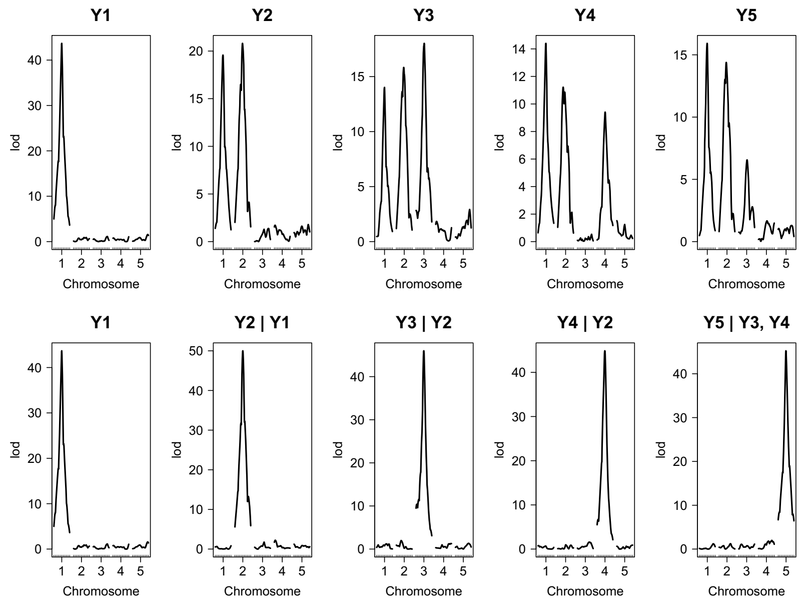
Although the structure of the phenotype network is correct, the genetic architecture is not.

Unconditional versus conditional QTL mapping

Here we plot the LOD profiles for all phenotypes using both unconditional mapping analysis, and conditional mapping (where the parents of each phenotype are used as additive covariates in the QTL mapping).

```
> par(mfrow = c(2, 5), cex.lab = 1.5, cex.axis = 1.5, cex.main = 2)
> uncond.nms <- paste("Y", 1:5, sep = "")
> for (i in 1:5) {
+   plot(Scan, lodcolumn = i, main = uncond.nms[i], ylab = "lod")
+ }
> plot(Scan, lodcolumn = 1, main = uncond.nms[1], ylab = "lod")
> cond.nms <- c("Y1", "Y2 | Y1", "Y3 | Y2", "Y4 | Y2", "Y5 | Y3, Y4")
> pheno.parents <- list(NULL, 1, 2, 2, c(3, 4))
> for (i in 2:5) {
+   CondScan <- scanone(Cross, pheno.col = i, method = "hk",
+                       addcov = Cross$pheno[, pheno.parents[[i]]])
+   plot(CondScan, main = cond.nms[i], ylab = "lod")
+ }
```

Unconditional versus conditional QTL mapping



QTLnet routines - basic functionality

Fit the QTLnet algorithm.

```
> out2 <- mcmc.qtlnet(cross = Cross,
+                     pheno.col = 1:5,
+                     threshold = 3.04,
+                     addcov = NULL,
+                     intcov = NULL,
+                     nSamples = 1000,
+                     thinning = 3,
+                     max.parents = 4,
+                     M0 = NULL,
+                     burnin = 0.2,
+                     method = "hk",
+                     random.seed = 987654321,
+                     init.edges = 0,
+                     saved.scores = NULL,
+                     rev.method = "nbhd",
+                     verbose = TRUE)
```

QTLnet routines - basic functionality

```
> summary(out2)

Model-averaged network: (min.prob = 0.5)
  cause effect prob
1    y1     y2    1
2    y2     y3    1
3    y2     y4    1
4    y3     y5    1
5    y4     y5    1

Posterior probabilities by direction:
   node1 node2   -->   <--    no
1     y1    y2 1.000 0.000 0.000
2     y1    y3 0.019 0.000 0.981
3     y1    y4 0.073 0.000 0.927
4     y1    y5 0.080 0.000 0.920
5     y2    y3 1.000 0.000 0.000
...

Acceptance frequency for MCMC: 0.9996667
```

QTLnet routines - basic functionality

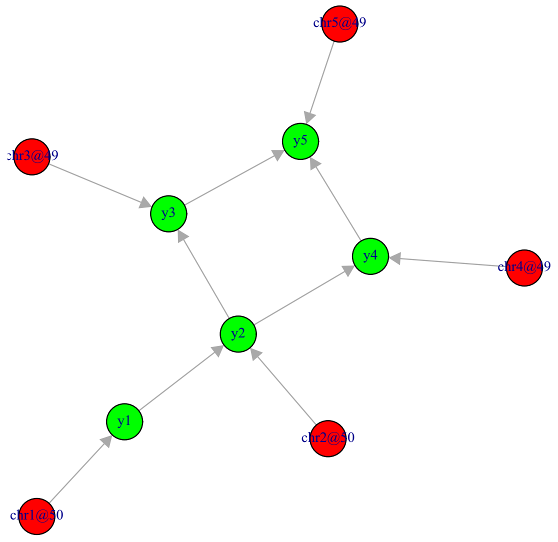
```
> print(out2)

Model averaged probabilities for edge direction (row -> col):
     [,1] [,2]  [,3]  [,4]  [,5]
[1,]    0    1 0.019 0.073 0.080
[2,]    0    0 1.000 1.000 0.094
[3,]    0    0 0.000 0.054 1.000
[4,]    0    0 0.029 0.000 1.000
[5,]    0    0 0.000 0.000 0.000

Posterior probabilities by causal model:
                                  post.prob      BIC
(1)(2|1)(3|2)(4|2)(5|3,4)       0.714107366 7149.930
(1)(2|1)(3|2)(4|2)(5|2,3,4)     0.081148564 7155.238
(1)(2|1)(3|2)(4|2,3)(5|3,4)     0.049937578 7156.117
(1)(2|1)(3|2)(4|2)(5|1,3,4)     0.037453184 7156.134
(1)(2|1)(3|2)(4|1,2)(5|3,4)     0.028714107 7154.531
(1)(2|1)(3|2,4)(4|2)(5|3,4)     0.027465668 7156.141
(1)(2|1)(3|2)(4|1,2)(5|1,3,4)   0.026217228 7160.734
...
```

QTLnet routines - basic functionality

```
> loci.qtlnet(out2)
$y1
[1] "chr1@50"

$y2
[1] "chr2@50"

$y3
[1] "chr3@49"

$y4
[1] "chr4@49"

$y5
[1] "chr5@49"
```

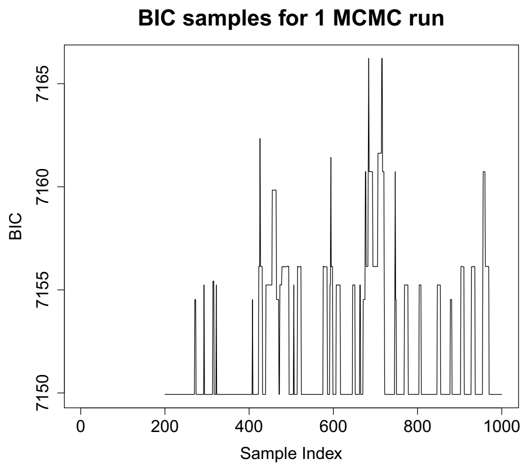
QTLnet routines - basic functionality

```
> plot(out2)
```



QTLnet routines - basic functionality

```
> par(mfrow = c(1, 1))
> plotbic.qtlnet(out2, smooth = FALSE)
```



QTLnet routines - parallel implementation

The most expensive part of calculations is running **scanone** on each phenotype with parent phenotypes as covariates. Our strategy is to pre-compute the BIC contributions using a cluster and save them for later use.

We divide the job into four steps:

1. Determine parents and divide into reasonable sized groups.
2. Compute BIC scores using scanone on a grid of computers.
3. Compute multiple MCMC runs on a grid of computers.
4. Catenate the outputs of the multiple MCMC runs into a single output object.

We illustrate this approach with a simple example of “parallel” analysis.

QTLnet routines - parallel implementation - step 1

STEP 1: defines how the computations are going to break up (that are carried out on steps 2 and 3).

```
> pheno.col <- 1:5
> max.parents <- 4
> size.qtlnet(pheno.col, max.parents)
[1] 80
> parents <- parents.qtlnet(pheno.col, max.parents)
> groups <- group.qtlnet(parents = parents, group.size = 10)
>
> save(Cross, pheno.col, max.parents, parents, groups,
+      file = "Step1.RData", compress = TRUE)
```

The function **size.qtlnet** determines the number of **scanone** calculations possible for a network with nodes **pheno.col** and maximum parent size **max.parents**.

```
> size.qtlnet(pheno.col, max.parents)
[1] 80
```

QTLnet routines - parallel implementation - step 1

The **parents.qtlnet** function creates a list of all possible parent sets (up to **max.parents** in size) to be used as covariates of the child phenotypes in the **scanone** computations.

The `parents` column shows the possible parent sets. The `n.child` column represents the number of possible child nodes to the parent set.

```
> parents <- parents.qtlnet(pheno.col, max.parents)
> parents
      parents n.child
                    5
1           1       4
2           2       4
...
1,2       1,2       3
...
```

No parents (5 `scanones`): $y_1 \sim 1$, $y_2 \sim 1$, $y_3 \sim 1$, $y_4 \sim 1$, and $y_5 \sim 1$.

With y_1 as a parent (4 `scanones`): $y_2 \sim y_1$, $y_3 \sim y_1$, $y_3 \sim y_1$, and $y_4 \sim y_1$.

With y_1 and y_2 as parents (3 `scanones`): $y_3 \sim y_1 + y_2$, $y_4 \sim y_1 + y_2$, and $y_4 \sim y_1 + y_2$.

QTLnet routines - parallel implementation - step 1

The function **group.qtlnet** groups the parent sets into roughly equal size groups for parallel computations.

```
> groups <- group.qtlnet(parents = parents, group.size = 10)
> groups
  begin end
1     1   2
2     3   4
3     5   7
4     8  10
5    11  14
6    15  18
7    19  23
8    24  30
9    31  31
> pa <- summary(parents)
> N <- rep(NA, nrow(groups))
> for (i in 1:nrow(groups))
+   N[i] <- sum(pa[seq(groups[i, 1], groups[i, 2]), 2])
> N
[1]  9  8 11  9 12 10 10 10  1
```

QTLnet routines - parallel implementation - step 2

STEP 2: Pre-compute BIC scores for selected parents.

```
> load("Step1.RData")
> for (i in seq(nrow(groups))) {
+   bic <- bic.qtlnet(Cross,
+                     pheno.col,
+                     threshold = 3.04,
+                     max.parents = max.parents,
+                     parents = parents[seq(groups[i,1], groups[i,2])])
+
+   save(bic, file = paste("bic", i, ".RData", sep = ""), compress = TRUE)
+ }
```

QTLnet routines - parallel implementation - step 2

Read in saved BIC scores and combine into one object.

```
> load("Step1.RData")
> bic.group <- list()
> for (i in seq(nrow(groups))) {
+   load(paste("bic", i, ".RData", sep = ""))
+   bic.group[[i]] <- bic
+   cat("group =", i, "\n")
+ }
> saved.scores <- bic.join(Cross, pheno.col, bic.group, max.parents = 4)
```

QTLnet routines - parallel implementation - step 2

```
> saved.scores
              y1        y2       y3       y4       y5
        1480.704 1785.9987 1982.565 2014.538 2677.213
1       1132.647 1414.8944 1776.324 1780.912 2437.802
2       1304.698 1222.2440 1394.943 1434.985 2005.474
3       1291.897 1242.0799 1682.636 1687.116 1953.474
4       1299.858 1156.7878 1273.851 1246.465 1917.227
1,2     1137.728 1059.9072 1400.437 1439.585 2011.442
1,3     1138.785 1089.4257 1627.265 1631.393 1917.990
1,4     1138.526 1023.7947 1276.038 1241.347 1885.897
2,3     1263.316 1002.2426 1401.154 1441.172 1800.790
2,4     1287.025 1110.6142 1221.812 1218.454 1759.518
3,4     1279.065 1128.8087 1210.769 1186.155 1424.405
1,2,3   1143.837  896.6137 1406.650 1445.789 1805.105
1,2,4   1143.942  984.3935 1225.789 1222.706 1765.651
1,3,4   1144.734 1000.8086 1202.754 1171.667 1430.608
2,3,4   1269.522 1008.2210 1091.324 1087.177 1429.712
1,2,3,4 1149.933  902.5707 1096.584 1093.126 1435.644
```

QTLnet routines - parallel implementation - step 3

STEP 3: Sample Markov chain (MCMC).

```
> set.seed(54321)
> n.runs <- 3
> for (i in seq(n.runs)) {
+   cat("run =", i, "\n")
+   ## Run MCMC with randomized initial network.
+   mcmc <- mcmc.qtlnet(Cross,
+                       pheno.col,
+                       threshold = 3.04,
+                       thinning = 1,
+                       max.parents = max.parents,
+                       saved.scores = saved.scores,
+                       init.edges = NULL)
+
+   save(mcmc, file = paste("mcmc", i, ".RData", sep = ""),
+        compress = TRUE)
+ }
```

QTLnet routines - parallel implementation - step 4

STEP 4: Combine results for post-processing.

```
> n.runs <- 3
> outs.qtlnet <- list()
> for (i in seq(n.runs)) {
+   load(paste("mcmc", i, ".RData", sep = ""))
+   outs.qtlnet[[i]] <- mcmc
+ }
> out3 <- c.qtlnet(outs.qtlnet)
```

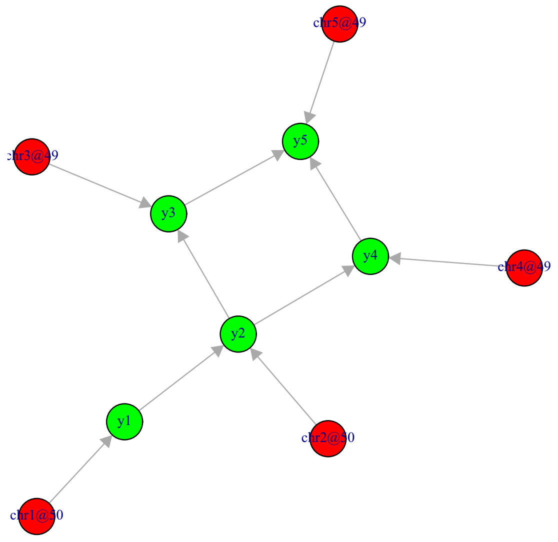
The function **c.qtlnet** catenates the outputs of the 3 separate runs together.

QTLnet routines - parallel implementation - outputs

```
> summary(out3)

Model-averaged network: (min.prob = 0.5)
  cause effect      prob
1    y1     y2 0.9155556
2    y2     y3 0.9255556
3    y2     y4 0.9129630
4    y3     y5 0.9085185
5    y4     y5 0.9103704

Posterior probabilities by direction:
   node1 node2   -->   <--    no
1     y1    y2 0.916 0.084 0.000
2     y1    y3 0.019 0.015 0.966
3     y1    y4 0.033 0.020 0.947
4     y1    y5 0.028 0.006 0.966
5     y2    y3 0.926 0.074 0.000
...

Acceptance frequency for MCMC: 0.999
```

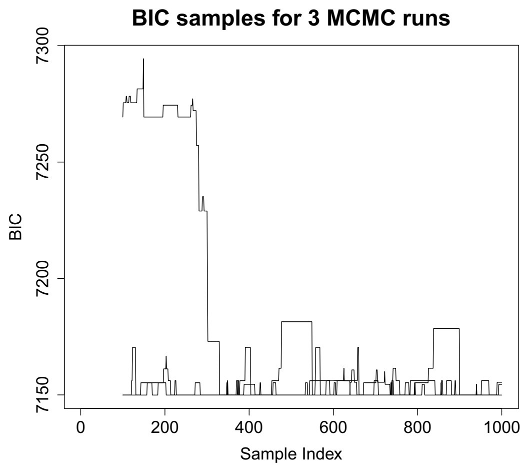
QTLnet routines - parallel implementation - outputs

```
> plot(out3)
```



QTLnet routines - parallel implementation - outputs

```
> plotbic.qtlnet(out3, smooth = FALSE)
```


BIC samples for 3 MCMC runs
BIC
Sample Index