Quantile-based permutation thresholds for QTL hotspot analysis: a tutorial

Elias Chaibub Neto* and Brian S Yandell†

May 30, 2012

1 Motivation

QTL hotspots, groups of traits co-mapping to the same genomic location, are a common feature of genetical genomics studies. Genomic locations associated with many traits are biologically interesting since they may harbor influential regulators. Nonetheless, non-genetic mechanisms, uncontrolled environmental factors and unmeasured variables are capable of inducing a strong correlation structure among clusters of transcripts, and as a consequence, whenever a transcript shows a spurious linkage, many correlated transcripts will likely map to the same locus, creating a spurious QTL hotspot. Permutation approaches that do not take into account the phenotypic correlation tend to underestimate the size of the hotspots that might appear by change in these situations (Breitling et al. 2008).

This issue motivated the development of permutation tests that preserve the correlation structure of the phenotypes in order to determine the significance of QTL hotspots (Breitling et al. 2008, Chaibub Neto et al. 2012). In this tutorial we present software tools implementing the NL-method (Chaibub Neto et al. 2012), the N-method (Breitling et al. 2008), and the Q-method (West et al. 2007, Wu et al. 2008) permutation approaches.

2 Overview

This tutorial illustrates the application of the NL-, N- and Q-methods, implemented in the `qtlhot` R package, to a few toy examples. The `qtlhot` package is built over the `R/qtl` package (Broman et al. 2003), and we assume the reader is familiar with it.

3 Basic functionality

In this section we consider two toy simulated examples. In the first we simulate highly correlated phenotypes. In the second, we simulate uncorrelated phenotypes.

*Department of Computational Biology, Sage Bionetworks, Seattle WA
†Department of Statistics, University of Wisconsin-Madison, Madison WI

```
> library(qtlhot)
```

We start by simulating a "null backcross" data set composed of 1,000 phenotypes, 4 chromosomes, 51 equally spaced genetic markers per chromosome, and 100 individuals, with the `sim.null.cross` function. The `latent.eff` parameter control the amount of correlation among the phenotypes. Each phenotype k is generated according to the model $Y_k = \theta L + \epsilon_k$, where $L \sim N(0, \sigma^2)$ is a latent variable, θ represents the effect (`latent.eff`) of the latent variable on the phenotype, and $\epsilon_k \sim N(0, \sigma^2)$ represents a residual error term with σ^2 set to `res.var`. Note that we do not simulate any QTLs in a "null cross" and any linkages we might detect in such a data set are due entirely to chance.

```
> 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)
```

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. Explicitly, it simulates each one of the hotspots according to the model $Y_k^* = \beta M + Y_k$, where Y_k is the phenotype generated by the `generate.null.cross` function; $M = \gamma Q + \epsilon_M$ is a master regulator that affects all phenotypes in the hotspot; Q is a QTL located at position `hpos` of chromosome `hchr`; γ represents the QTL effect (`Q.eff`); $\epsilon_M \sim N(0, \sigma^2)$; and β is computed such that the association between Y_k^* and Q, measured by the LOD score, is given (theoretically) by a valued sampled from the user specified LOD score range (`lod.range.1` and etc).

```
> 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)
```

Note that by choosing `latent.eff = 3` we generate highly correlated phenotype data. The distribution of the correlation values for each pair of phenotypes is given below.

```
> 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
```

Next we obtain standard permutation thresholds (Churchill and Doerge 1994) for single trait QTL mapping analysis for the sequence `alphas`, representing target genome wide error rates (GWER).

```
> 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)
```

We perform QTL mapping analysis for all 1,000 phenotypes using Haley-Knott regression, and process the LOD profiles by setting to zero LOD values outside the 1.5 LOD support interval (Manichaikul et al. 2006) around the peak at each chromosome (as well as LOD values below the single trait mapping threshold, `thr`). LOD support intervals are the most commonly used interval estimate for the location of a QTL. By setting to zero the LOD scores outside the LOD support interval we can considerably decrease the spread of the hotspot.

```
> 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)
```

Next we infer the hotspot architecture at varying QTL mapping thresholds. In other words, for each genomic position, we count the number of traits that map to it with a LOD score equal or higher than the threshold in `lod.thrs`. The `counts1` object is a matrix with 204 rows representing the genetic markers, and 10 columns representing the varying QTL mapping thresholds. As an illustration, we show the counts for the 5 first markers on chromosome 2. The first column gives the counts associated with QTL mapping threshold of 3.11, whereas the last one shows the counts based on the more liberal threshold 2.11. Note how the counts increase as the QTL mapping thresholds decrease.

```
> 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
```

We plot the hotspot architecture inferred using the single trait permutation threshold 2.44 ($\alpha = 0.05$). Figure 1 shows the counts across the genome. Recall that in the call of function `include.hotspots` we set to simulate 3 hotspots: (1) a hotspot of size 100 at position 25cM of chromosome 2 with LOD scores around 2.5; (2) a hotspot of size 50 at position 75cM of chromosome 3 with LOD scores ranging from 5 to 8; and (3) a hotspot of size 20 at position 50cM of chromosome 4 with LOD scores ranging from 10 to 15. Nonetheless, Figure 1 shows several spurious peaks on chromosome 1, that arise because of the high correlation of the phenotypes.

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

Next, we perform permutation tests to assess the statistical significance of the hotspots detected on Figure 1. We start with the Q-method permutations. The `WW.perm` function implements the Q-method's permutation scheme (see the Method's section of Chaibub Neto et a. 2012, for details). The `n.perm` parameter specifies the number of simulations. Here we set it to 100 in order to save time. In practice, we recommend at least 1,000 permutations. The function's output is a matrix with 100 rows representing the permutations, and 10 columns representing the QTL mapping thresholds. Each entry ij, represents the maximum number of significant linkages across the entire genome detected at permutation i, using the LOD threshold j. The `WW.summary` function computes the Q-method's hotspot size permutation thresholds, that is, the $1 - \alpha$ quantiles for each one of the QTL mapping LOD thrsholds in `lod.thrs`. For instance, the entry at row 10 and column 1 of the `Q.1.thr` matrix tells us that the 99% percentile of the permutation distribution of genome wide maximum hotspot size based on a QTL mapping threshold of 2.11 is 27.00. In other words, any hotspot greater than 27 is considered statistically significant at a 0.01 significance level when QTL mapping is done using a 2.11 LOD threshold.

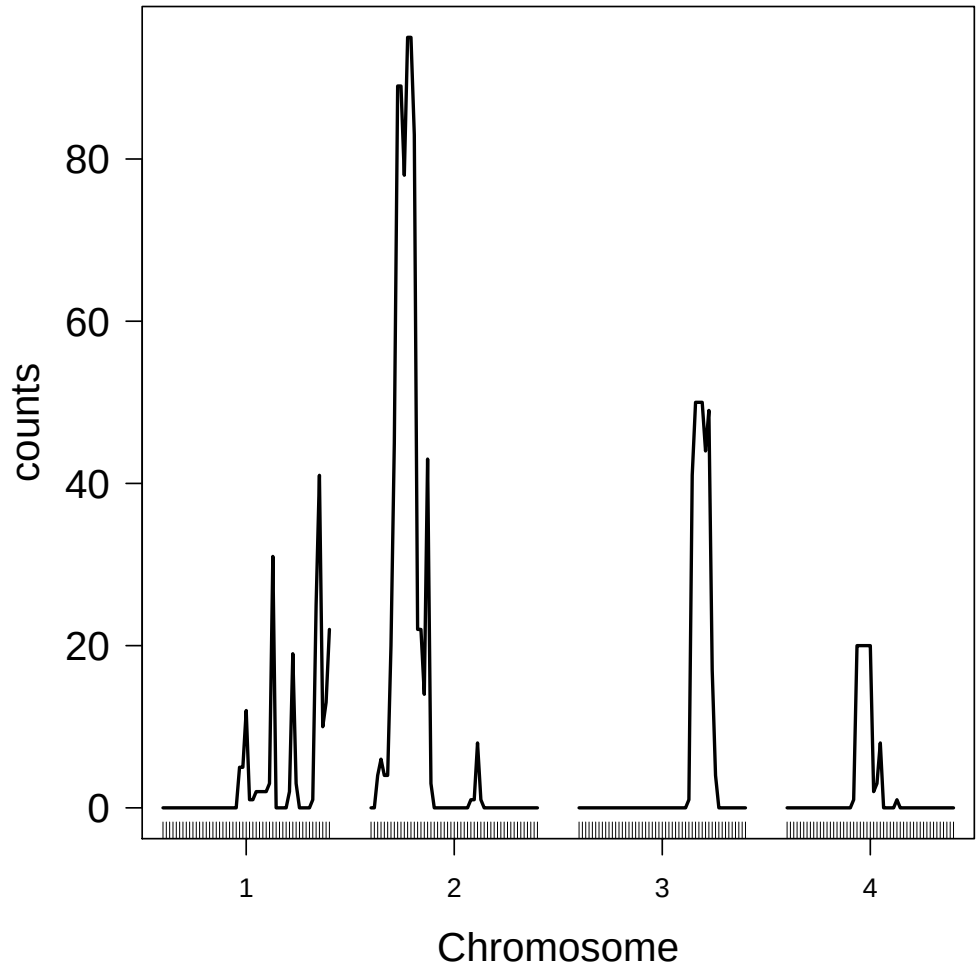

Figure 1: Hotspot architecture associated with QTL mapping threshold of 2.44 in example 1.

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

> 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
2.89135162173149 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
2.25777470881538 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
2.150234395168   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
```

In general, we are often interested in using the same error rates for the QTL mapping and hotspot analysis. That is, if we adopt a QTL mapping threshold that controls GWER at a 1% level (in our case, 3.11) we will also want to consider $\alpha = 0.01$ for the hotspot analysis, leading to a hotspot threshold of 12.00. Therefore, we are usually more interested in the diagonal of "Q.1.thr". For the hotspots depicted in Figure 1, we adopted a GWER of 5%, and the corresponding Q-method's permutation threshold is 18. According to this threshold, all hotspots on Figure 1 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
```

Next we consider the N- and NL-methods. The `NL.N.perm` function implements the N- and NL-methods' permutation schemes (see Chaibub Neto et al. 2012, for details). The parameter `Nmax` sets the maximum hotspot size to be analyzed by the NL-method. The parameter `drop` controls the magnitude of the LOD support interval computation during the LOD profile processing step. The function's output is a list with two elements: `max.lod.quant` and `max.N`.

```
> set.seed(12345)
> NL.N.1 <- NL.N.perm(cross = cross1,
+                     Nmax = 300,
+                     n.perm = 100,
+                     lod.thrs = lod.thrs,
+                     drop = 1.5,
+                     verbose = FALSE)

> names(NL.N.1)

[1] "max.lod.quant" "max.N"
```

The `max.lod.quant` object stores the output of the NL-method's permutations and is given by a matrix with 100 rows representing the permutations, and 300 columns representing the hotspot sizes analyzed. Entry ij stores the maximum genome wide $qLOD(n)$ value computed at permutation i using the QTL mapping threshold j. The statistic $qLOD(n)$ corresponds to the nth LOD score in a sample ordered from highest to lowest. For instance, consider the first 10 lines and 8 columns of `max.lod.quant`. At the 6th permutation, we have that the maximum LOD score across the genome is 3.58, the second maximum across the genome is 3.55, and so on.

```
> NL.N.1[[1]][1:10, 1:8]
```

```
             1        2        3        4        5        6        7        8
 [1,] 2.115918 1.903466 1.713409 1.649016 1.600378 1.594265 1.587357 1.557915
 [2,] 2.464650 2.162832 1.932474 1.885934 1.878833 1.839507 1.827621 1.815150
 [3,] 3.374947 3.358949 3.198482 3.195974 3.121577 3.105578 3.028484 3.026301
 [4,] 2.884215 2.867459 2.660496 2.647943 2.560766 2.547804 2.533727 2.510926
 [5,] 4.188665 3.759857 3.664104 3.656594 3.560914 3.523829 3.517086 3.497425
 [6,] 3.549040 3.360962 3.352198 3.252716 3.219246 3.174312 3.170193 3.147144
 [7,] 2.703879 2.494766 2.377396 2.259549 2.206756 2.127329 2.123880 2.114704
 [8,] 3.239116 3.229937 3.158906 3.074346 3.006857 3.004526 2.995804 2.991072
 [9,] 3.032630 3.008468 2.855149 2.771142 2.746501 2.685583 2.656720 2.638065
[10,] 3.199169 3.006626 2.981184 2.923865 2.914737 2.871451 2.780572 2.776395
```

The `max.N` stores the output of the N-method's permutations and is given by a matrix with 100 rows representing the permutations, and 10 columns representing the QTL mapping thresholds. Entry ij stores the maximum genome wide hotspot size detected at permutation i using the QTL mapping threshold j. For illustration we show it's first 12 lines (note that we are transposing the output).

```
> t(NL.N.1[[2]][1:12,])
```

```
                 [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12]
3.10508056313925    0    0    6    0   19    9    0    3    0     1     0     3
2.89135162173149    0    0   14    0   31   18    0   14    2     5     0     4
2.67690269000741    0    0   26    2   45   34    1   36    6     9     0     9
2.5743266994317     0    0   40    4   52   60    1   55   17     9     0    11
2.43869721183317    0    1   65   13   66   97    2  100   25    14     1    15
2.335067939838      0    1   83   25   81  158    3  151   32    17     1    17
2.25777470881538    0    1  106   36   90  213    4  191   40    18     1    22
2.19884780562269    0    1  131   46  101  249    5  224   50    18     1    24
2.150234395168      0    2  162   61  116  290    5  254   61    19     1    25
2.11039422475441    1    2  186   75  127  328    8  281   64    19     1    28
```

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]]
```

The `N.1.thr` object is a 10 by 10 matrix with rows indexing the QTL mapping thresholds and columns indexing the target genome wide error rates. Each entry ij shows the hotspot size above which a hotspot is considered significant at a GWER j using the QTL mapping threshold i. As before, our interest focus on the diagonal, and 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 200.55. Note that according to the N-method, none of the hotspots on Figure 1 is significant.

```
> N.1.thr

                    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
2.89135162173149  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
2.5743266994317  249.59 239.14 204.08 138.68 129.35  93.28  79.84  78.08  63.44
2.43869721183317 352.30 309.80 275.05 212.56 195.75 144.24 127.98 121.48 101.89
2.335067939838   432.25 389.80 354.08 286.76 263.10 191.50 176.77 173.24 159.35
2.25777470881538 490.01 445.84 407.17 350.32 322.35 240.16 228.49 215.12 213.09
2.19884780562269 527.81 491.68 462.87 402.48 373.40 288.34 266.26 250.28 240.81
2.150234395168   562.64 531.58 499.96 443.32 413.45 321.76 295.54 290.32 272.71
2.11039422475441 604.35 565.74 540.75 473.76 444.40 350.88 334.77 328.48 303.43
                    0.1
3.10508056313925  15.3
2.89135162173149  25.4
2.67690269000741  45.1
2.5743266994317   60.2
2.43869721183317  98.2
2.335067939838   151.7
2.25777470881538 193.2
2.19884780562269 225.6
2.150234395168   262.0
2.11039422475441 286.6

> 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
```

The `NL.1.thr` object is a matrix with 300 rows representing the spurious hotspot sizes analyzed, and 10 columns representing the target genome wide error rates. 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. As a illustration we show the first five lines.

```
> round(NL.1.thr[1:5,], 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
4 4.2988 4.1414 4.1100 3.7739 3.6618 3.6439 3.5743 3.4809 3.4273 3.4215
5 4.2838 4.1040 4.0347 3.6636 3.5653 3.5543 3.5462 3.4668 3.3749 3.3165
```

To visualize these results, we plot on Figure 2 the hotspot significance profile for the thresholds targeting GWER at a 5% level.

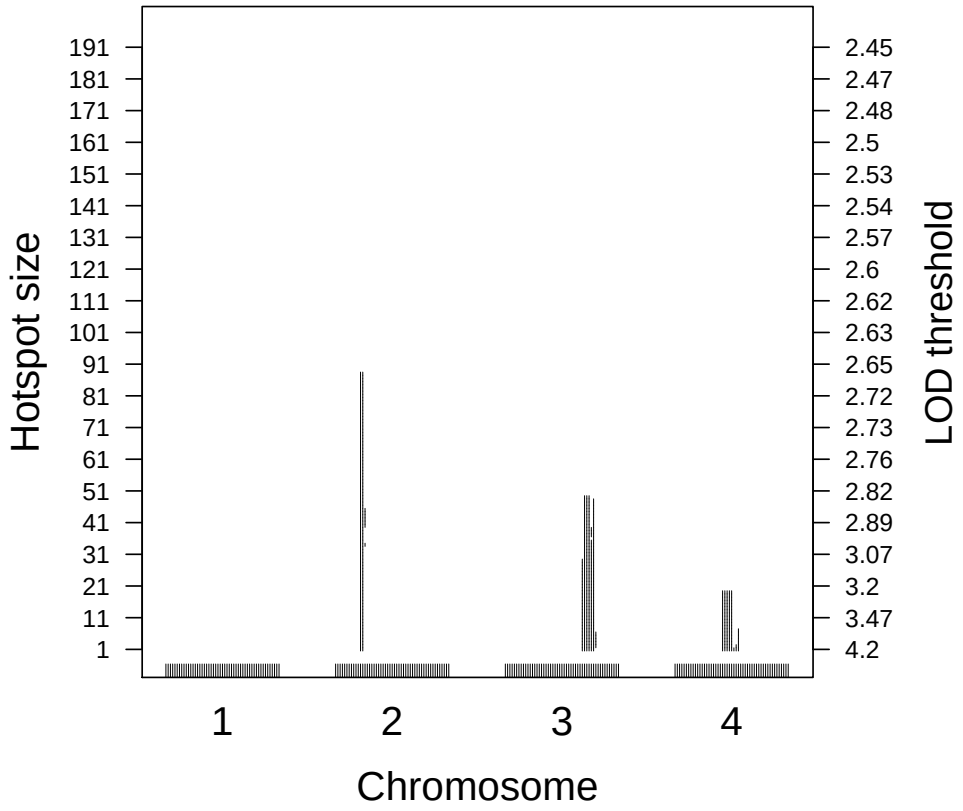

Figure 2: Hotspot size significance profile targeting GWER at a 5% level for example 1.

```
> N.1 <- round(N.1.thr[5, 5])
. 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))
```

Figure 2 depicts a sliding window of hotspot size thresholds ranging from $n = 1, \ldots, N$, where $N = 201$ corresponds to the (approximate) hotspot size threshold derived from the N-method. 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. For example, the hotspot on chromosome 3 was significant up to size 20, meaning

that more than 1 trait mapped to the hotspot locus with LOD higher than 3.99, more than 11 traits mapped to the hotspot locus with LOD higher than 3.36, and so on up to hotspot size 49 where more than 49 traits mapped to the hotspot locus with LOD higher than 2.84.

Contrary to the Q-method that detected all the hotspots on Figure 1 as significant (including the spurious hotspots on chromosome 1), and the N-method that did not detect any hotspots, the NL-method's sliding window correctly detected the simulated hotspots and showed that the apparent hotspots on chromosome 1 were noisy artifacts.

4 Example with Uncorrelated Phenotypes

Next we consider a second toy example with uncorrelated phenotype data. We repeat the simulation and analysis steps presented previously changing `latent.eff` to zero.

```
> ncross2 <- sim.null.cross(chr.len = rep(100,4),
+                           n.mar = 51,
+                           n.ind = 100,
+                           type = "bc",
+                           n.pheno = 1000,
+                           latent.eff = 0,
+                           res.var = 1,
+                           init.seed = 123457)
> cross2 <- include.hotspots(cross = ncross2,
+                            hchr = c(2, 3, 4),
+                            hpos = c(25, 75, 50),
+                            hsize = c(100, 50, 20),
+                            Q.eff = 2,
+                            latent.eff = 0,
+                            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)
> nphe2 <- as.matrix(cross2$pheno)
> ncor2 <- cor(nphe2)
> ncor2 <- ncor2[lower.tri(ncor2)]
> summary(ncor2)

     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
-0.471200 -0.067160  0.001263  0.002224  0.070240  0.666900

> scan2 <- scanone(cross2, pheno.col = 1:1000, method = "hk")
> scandrop2 <- set.to.zero.beyond.drop.int(chr = scan2[,1],
+                                          scanmat = as.matrix(scan2[,-c(1,2)]),
```

```
+                                          thr = min(lod.thrs),
+                                          drop = 1.5)
> counts2 <- t(count.thr(scandrop2, lod.thrs, droptwo=FALSE))
> out2 <- data.frame(scan2[, 1:2], counts2)
> class(out2) <- c("scanone", "data.frame")

> par(mar=c(4.1,4.1,0.1,0.1))
> plot(out2, lodcolumn = 5, ylab = "counts", cex.lab = 1.5, cex.axis = 1.5)
```

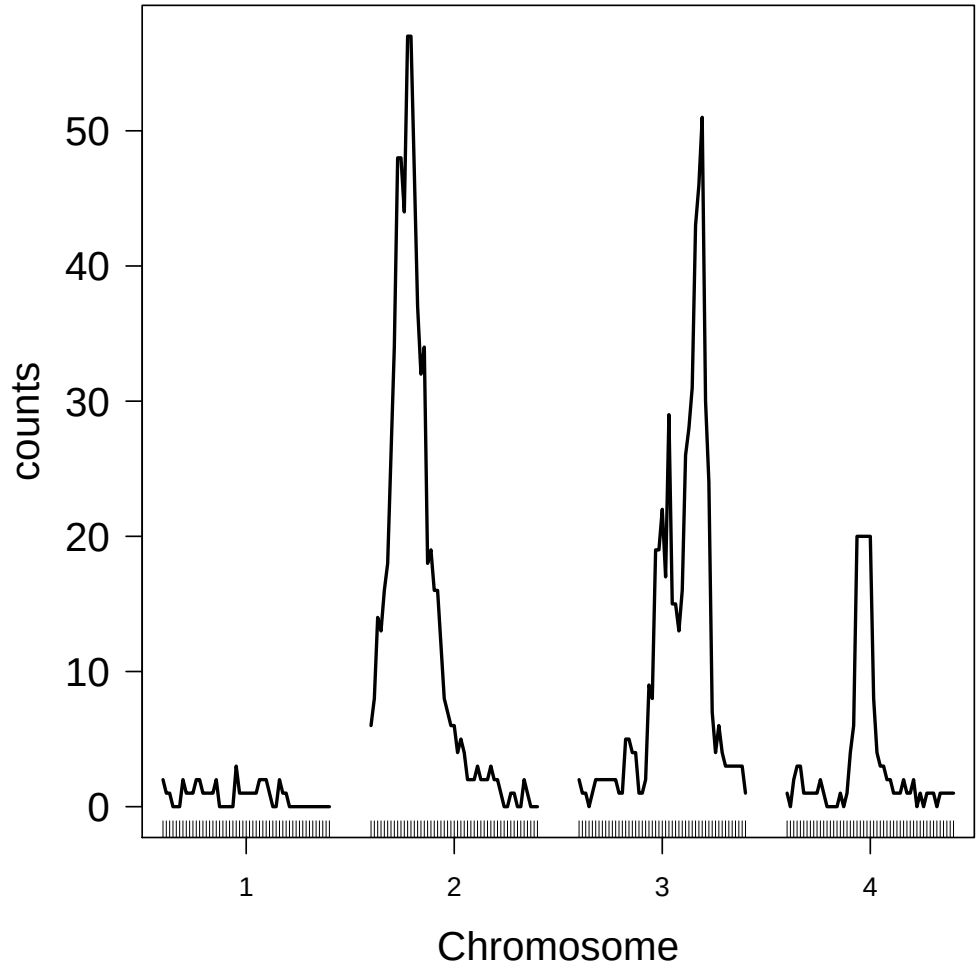

Figure 3: Hotspot architecture targeting 5% GWER for example 2.

```
> set.seed(12345)
> Q.2 <- WW.perm(scanmat = scandrop2,
+                lod.thrs = lod.thrs,
```

```
+                  n.perm = 100,
+                  verbose = FALSE)

> Q.2.thr <- WW.summary(Q.2, alphas)

> set.seed(12345)
> NL.N.2 <- NL.N.perm(cross = cross2,
+                       Nmax = 300,
+                       n.perm = 100,
+                       lod.thrs = lod.thrs,
+                       drop = 1.5,
+                       verbose = FALSE)

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

The Q-method thresholds are quite similar to the previous example. This is not unexpected since the number of significant linkages detected in examples 1 and 2 were similar (respectively, 1438 and 1413 for a LOD threshold of 2.44), and the Q-method thresholds are a function of the number of significant QTLs (the higher the number of significant linkages, the higher the threshold) and not of the correlation among the phenotypes.

```
> diag(Q.2.thr)

 [1] 14.00 14.02 15.03 17.00 17.00 18.00 18.00 18.08 19.00 19.00

> apply(counts1, 2, sum)

 [1]  590  752  980 1129 1350 1591 1814 2013 2210 2389

> apply(counts2, 2, sum)

 [1]  944 1065 1234 1312 1440 1540 1610 1678 1760 1811
```

The N-method, as expected, gave rise to much smaller thresholds in this second example with uncorrelated phenotypes. Additionally, inspection of Figure 3 shows no spurious hotspots on chromosome 1.

```
> diag(N.2.thr)

 [1] 4.00 4.02 5.00 6.00 6.00 7.00 7.00 8.00 8.00 8.10

> N.2 <- round(N.2.thr[5, 5], 0)
> sliding.bar.plot(scan = data.frame(scan2[, 1:2], scandrop2),
+                   lod.thr = NL.2.thr[1:N.2, 5],
+                   size.thr = 1:N.2,
+                   gap = 50,
+                   y.axes = seq(1, N.2, by = 1))
```

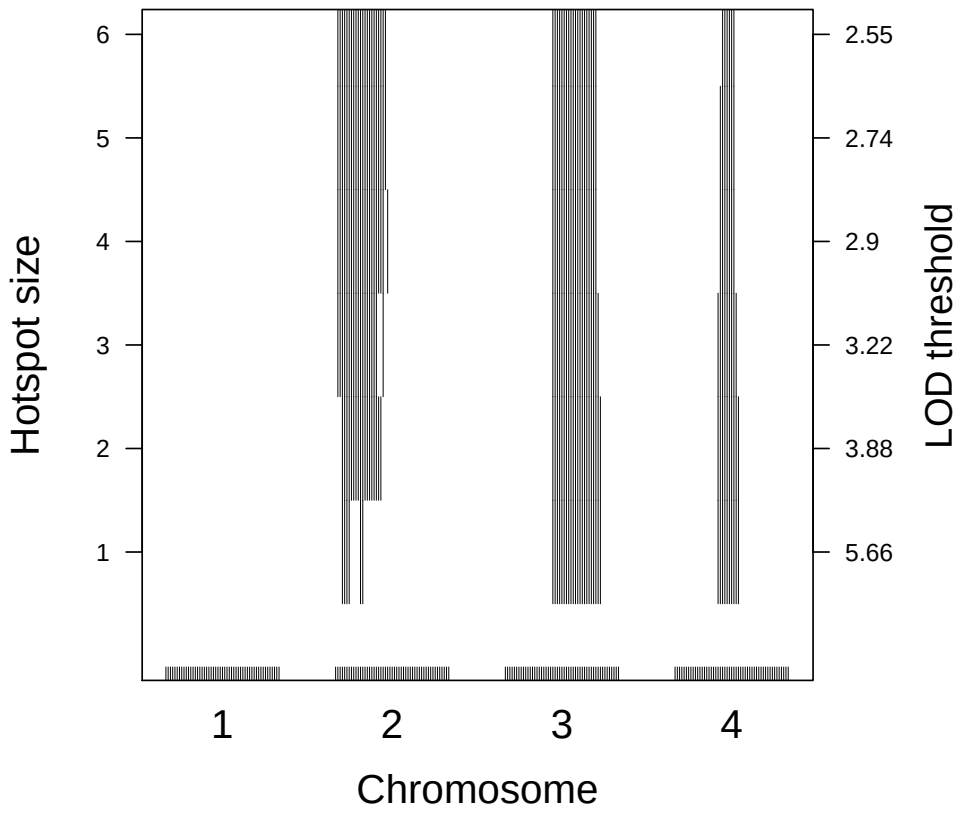

Figure 4: Hotspot significance profile targeting 5% GWER for example 2.

Figure 4 presents the hotspot significance profile targeting 5% GWER. For this second example, all methods correctly detected the simulated hotspots.

```
> hot.scan2 <- qtlhot.scan(cross2, scan2, NL.N.2$max.lod.quant, lod.thrs, probs = seq(.01, .1,
> par(mar = c(4.1,4.1,0.1,4.1))
> plot(out2, lodcolumn = 5, ylab = "counts", cex.lab = 1.5, cex.axis = 1.5)
> plot(hot.scan2, lodcolumn = 2, add = TRUE, col = "red")
> ## Add right axis with
> quant <- attr(hot.scan2, "quant")
> tmp <- seq(along = quant)
> axis(4, at = tmp, label = round(quant[tmp], 2), las = 1, cex = 0.35)
> mtext("sliding LOD thresholds",4, 1, cex=1.5)
```

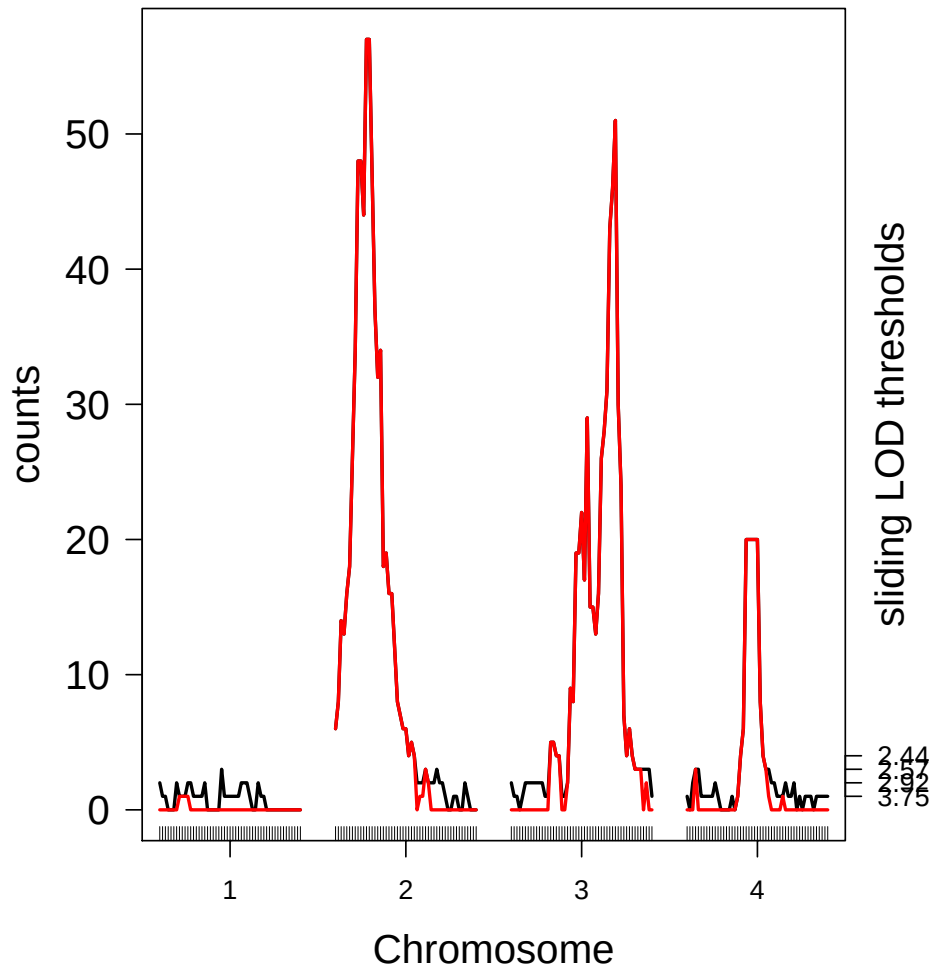

Figure 5: Hotspot significance scan targeting 5% GWER for example 2.

Figure 5 shows another way to represent significant hotspots. We overlay the largest significant hotspot counts using the sliding quantiles in red on top of the curve on Figure 3. Notice that the large sizes are all significant, but only small sizes corresponding to larger LOD scores are significant. We add a right axis with the sliding LOD thresholds.

5 References

1. Breitling R., Y. Li, B. M. Tesson, J. Fu, C. Wu, et al., 2008 Genetical genomics: spotlight on QTL hotspots. PLoS Genetics **4:** e1000232.

2. Broman K. W., W. Wu, S. Sen, G. A. Churchill, 2003 R/qtl: QTL mapping in experimental crosses. Bioinformatics **19**: 889-890.

3. Chaibub Neto et al., 2012 Quantile-based permutation thresholds for QTL hotspots. Genetics (under review).

4. Churchill G. A., and R. W. Doerge, 1994 Empirical threshold values for quantitative trait mapping. Genetics **138**: 963-971.

5. Manichaikul A., J. Dupuis, S. Sen, and K. W. Broman, 2006 Poor performance of bootstrap confidence intervals for the location of a quantitative trait locus. Genetics **174:** 481-489.

6. West M. A. L., K. Kim, D. J. Kliebenstein, H. van Leeuwen, R. W. Michelmore, R. W. Doerge, D. A. St. Clair 2007 Global eQTL mapping reveals the complex genetic architecture of transcript-level variation in Arabidopsis. Genetics **175:** 1441-1450.

7. Wu C., D. L. Delano, N. Mitro, S. V. Su, J. Janes, et al. 2008 Gene set enrichment in eQTL data identifies novel annotations and pathway regulators. PLoS Genetics **4:** e1000070.