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**.

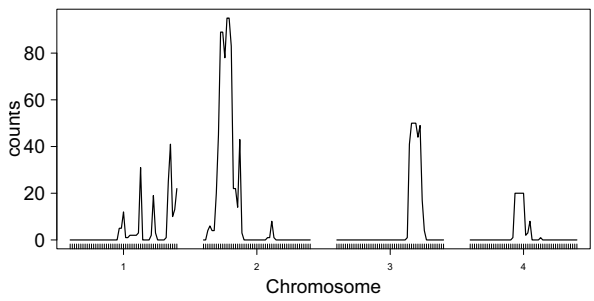
The counts1 object is a matrix with 204 rows (genetic markers) and 10 columns (thresholds).

```
> 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 *n*th 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.