R/qtl Demo

Tyler J Reich

2025-12-12

Preface on this Project:

With almost no background in computer science, learning to write code has been a significant challenge; however, it is one I am eager to undertake. I am grateful that we are allowed to use generative AI in this class, as ChatGPT has been an invaluable resource in helping me develop and understand the R/qtl workflow for QTL mapping. While ChatGPT assisted me in generating the R code to import, process, and analyze genotype and phenotype data, I also asked it to explain what each line of code does and why it is used in the context of QTL analysis. I am prepared to explain this workflow to the best of my ability in order to demonstrate my understanding of how and why it works.

Preparing Variant Data for R/qtl Analysis

This section describes how to convert genomic Variant Call Format (gVCF/VCF) files from the GATK pipeline into a format compatible with R/qtl for QTL mapping.

Requirements:

1. R (>=4.2 recommended)
2. R/qtl package
3. bcftools
4. vcftools (optional, for filtering or subsetting VCFs) 5.CSV conversion script (provided in scripts/)

---

1. Obtain VCF Files from the GATK Pipeline

After low-coverage whole-genome sequencing (lcWGS), single-sample gVCFs are generated with **GATK HaplotypeCaller** in reference-confidence mode. These are then combined and genotyped into a multi-individual VCF using:

- `GenomicsDBImport`
- `GenotypeGVCFs`

This final VCF (or one subset for testing) serves as the input for conversion to R/qtl format.

---

2. Convert VCF to CSV for R/qtl and Add Phenotype Data

R/qtl expects a single CSV file in which:

- **Each row = one individual**
- **Columns = individual ID, phenotype values, followed by marker genotypes**

Because VCFs are marker-major (one row per variant), they must be transposed into individual-major format.

Extract genotypes using bcftools

Use `bcftools query` to extract marker positions and genotypes:

Extract marker positions and per-individual genotypes from a VCF file

```
(bash) bcftools query -f '%CHROM\t%POS[\t%GT]\n' input.vcf.gz > genotypes_raw.tsv

Notes:
bcftools query -> Calls the query command in bcftools, which extracts specific fields from a VCF using a
-f '%CHROM\t%POS[\t%GT]\n' -> The format string that tells bcftools exactly what information to print:
%CHROM → chromosome of each variant
%POS → genomic position of the variant
[\t%GT] → for each sample in the VCF, print a tab (\t) followed by the genotype (%GT)
The square brackets indicate "repeat this for each sample."
\t → literal tab separators
\n → end the line after each variant
```

Each output row will look like: chrom pos sample1_GT sample2_GT sample3_GT ...

input.vcf.gz -> The compressed VCF file produced by GATK after joint genotyping. genotypes_raw.tsv -> Redirects the output into a tab-separated text file named genotypes_raw.tsv.

This produces: 1. One row per marker 2. One column per individual 3. Genotypes reported in VCF format (e.g., 0/0, 0/1, 1/1)

Convert to R/qtl format: A provided script (in scripts/) converts this marker-major file into R/qtl's individual-major CSV format. This script: 1. Transposes the genotype matrix 2. Converts VCF genotypes (0/1) to R/qtl genotypes (e.g., AB) 3. Adds marker names and chromosome positions 4. Ensures consistent delimiters and header formatting

Add phenotypes: Before loading into R/qtl, phenotype measurements must be included as columns immediately after the id column.

Once combined, this CSV can be imported directly into R/qtl using:

```
cross <- read.cross(format = "csv", file = "filename.csv")
```

Preface for the R/qtl Analysis Section

Before applying the QTL mapping pipeline to my Poecilia F2 dataset, I validated all downstream analytical steps using a built-in example dataset provided by R/qtl. The `listeria` dataset is an experimentally derived F2 intercross with known genotypes and phenotypes (Broman et al., 2003), making it an ideal proof-of-concept system. The following analysis demonstrates that the entire R/qtl workflow from data import and quality control to genome scans, permutation testing, confidence interval estimation, and effect-size visualization—runs successfully on a complete and well-formatted F2 dataset. Once my Poecilia genotypes (VCFs -> CSV) and phenotypes are generated, only the input files will need to be replaced, and the same workflow can be directly applied.

Using Built-In Example Dataset from R/qtl

This section begins by loading the listeria dataset, a built-in F2 intercross included with the R/qtl package. Importing the dataset into the object cross allows all downstream QTL-mapping functions, such as quality

control, genome scanning, permutation testing, and effect-size visualization, to be demonstrated on a fully-formatted and validated example dataset.

```
library(qtl)
```

```
## Warning: package 'qtl' was built under R version 4.3.3
```

```
# Load the listeria F2 intercross dataset
data(listeria)
cross <- listeria
```

Inspect the Cross

This step inspects the structure of the imported cross object to confirm that the dataset contains the expected phenotypes, chromosomes, and genetic markers. The `summary(cross)` command provides an overview of the cross type, number of individuals, phenotypic traits, markers, and missing data, while `names(cross$pheno)` and `names(cross$geno)` list the available phenotype columns and chromosomes, respectively. This ensures the dataset is correctly loaded and ready for QTL analysis.

```
summary(cross)
```

```
##     F2 intercross
##
##     No. individuals:    120
##
##     No. phenotypes:     2
##     Percent phenotyped: 96.7 100
##
##     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:      133
##     No. markers:        13 6 6 4 13 13 6 6 7 5 6 6 12 4 8 4 4 4 4 2
##     Percent genotyped:  88.5
##     Genotypes (%):
##           Autosomes:        CC:25.8      CB:48.9      BB:24.4  not BB:0.0
##                         not CC:0.9
##        X chromosome:        CC:51.7      CB:48.3
```

```
# Check phenotypes and genotypes
names(cross$pheno)   # phenotypes: "T264", "sex"
```

```
## [1] "T264" "sex"
```

```
names(cross$geno)    # chromosomes: 1, 2, ..., 19, X
```

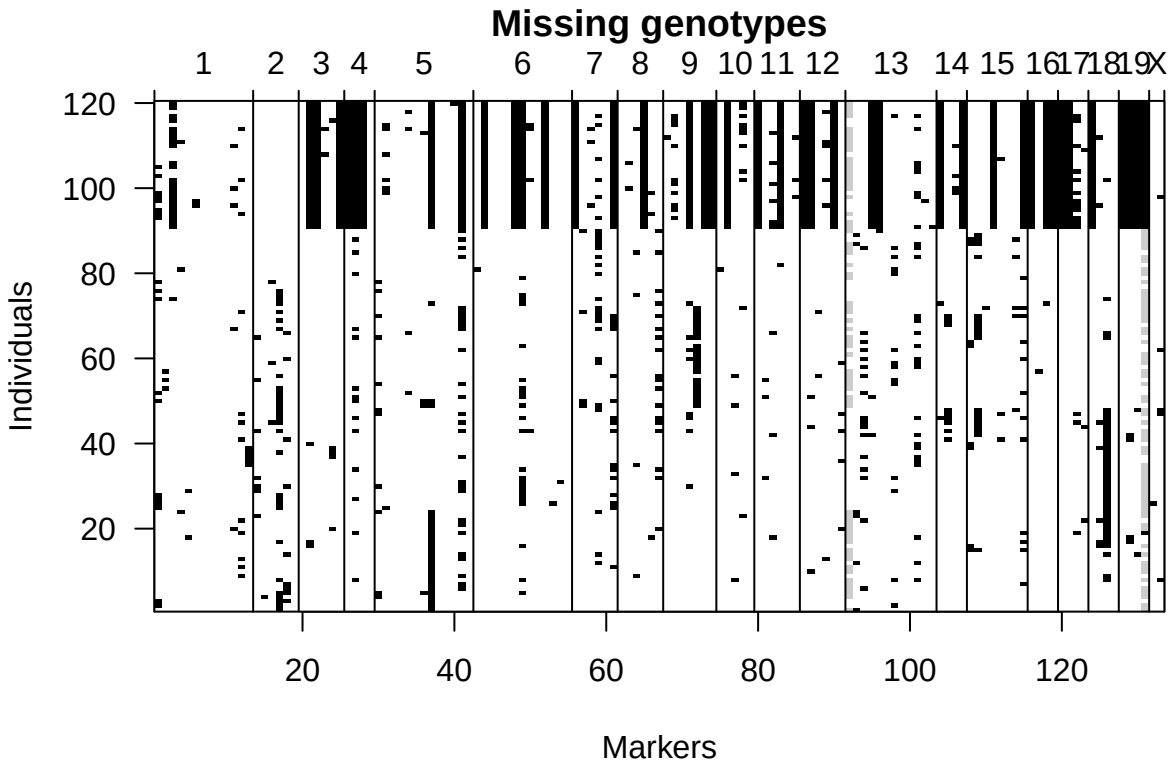
```
##  [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10" "11" "12" "13" "14" "15"
## [16] "16" "17" "18" "19" "X"
```

Quality Control Checks

These quality-control checks evaluate data completeness and marker reliability before performing QTL mapping. The `plotMissing(cross)` function visualizes missing genotype data across individuals and markers, helping identify potential issues such as poorly performing markers or samples with excessive missingness. The `geno.table(cross)` command summarizes genotype frequencies for each marker, allowing assessment of

segregation distortion or unexpected genotype ratios. Together, these checks ensure the dataset meets quality standards for accurate downstream QTL analysis.

```
# Plot missing data
plotMissing(cross)
```




```
# Check segregation ratios
geno.table(cross)
```

```
##          chr missing CC CB BB not.BB not.CC      P.value
## D10M44     1      19 29 46 26      0      0 0.6125657411
## D1M3       1       3 26 60 31      0      0 0.7771384555
## D1M75      1      24 15 52 29      0      0 0.0930144892
## D1M215     1       3 22 63 32      0      0 0.3009368909
## D1M309     1       2 31 52 35      0      0 0.3805637036
## D1M218     1       2 30 52 36      0      0 0.3212315037
## D1M451     1       0 29 57 34      0      0 0.6988400895
## D1M504     1       0 31 55 34      0      0 0.6116062006
## D1M113     1       0 30 57 33      0      0 0.7985162188
## D1M355     1       0 30 57 33      0      0 0.7985162188
## D1M291     1       5 26 59 30      0      0 0.8367241192
## D1M209     1      12 26 57 25      0      0 0.8386801055
## D1M155     1       5 26 59 30      0      0 0.8367241192
## D2M365     2       7 24 61 28      0      0 0.6065306597
## D2M37      2       1 29 60 30      0      0 0.9874740688
## D2M396     2       3 27 62 28      0      0 0.8041666799
## D2M493     2      30 24 44 22      0      0 0.9355069850
```

```
## D2M226    2       9 33 54 24      0      0 0.4628879466
## D2M148    2       0 34 62 24      0      0 0.4065696597
## D3M265    3       0 33 66 21      0      0 0.1652988882
## D3M51     3      33 26 46 15      0      0 0.2155671626
## D3M106    3      30 28 45 17      0      0 0.2606844924
## D3M257    3       2 33 69 16      0      0 0.0158582121
## D3M147    3       5 29 69 17      0      0 0.0286622626
## D3M19     3      30 23 50 17      0      0 0.3845984194
## D4M2      4      30 18 49 23      0      0 0.5308194506
## D4M178    4      44 12 46 18      0      0 0.1155681665
## D4M187    4      30 13 48 29      0      0 0.0476227617
## D4M251    4       0 22 62 36      0      0 0.1826835241
## D5M148    5      11 27 47 35      0      0 0.1980439991
## D5M232    5       7 26 53 34      0      0 0.4569479266
## D5M257    5       0 32 55 33      0      0 0.6537697851
## D5M83     5       0 31 56 33      0      0 0.7408182207
## D5M307    5       4 32 53 31      0      0 0.6442585409
## D5M357    5       0 31 58 31      0      0 0.9355069850
## D5M205    5       4 27 56 33      0      0 0.6843332004
## D5M398    5      57 15 30 18      0      0 0.8071177470
## D5M91     5       0 27 59 34      0      0 0.6537697851
## D5M338    5       0 27 61 32      0      0 0.7985162188
## D5M188    5       1 28 57 34      0      0 0.6652695213
## D5M29     5      58 13 25 24      0      0 0.0444716565
## D5M168    5       0 30 52 38      0      0 0.2018965180
## D6M223    6       1 32 59 28      0      0 0.8705279066
## D6M188    6      30 21 54 15      0      0 0.1108031584
## D6M284    6       0 27 76 17      0      0 0.0060967466
## D6M39     6       0 24 75 21      0      0 0.0218184355
## D6M254    6       0 18 77 25      0      0 0.0053803600
## D6M194    6      30 13 54 23      0      0 0.0544152349
## D6M290    6      53 13 35 19      0      0 0.5463597639
## D6M25     6       4 20 67 29      0      0 0.1230914645
## D6M339    6       0 22 68 30      0      0 0.2018965180
## D6M59_    6      30 16 49 25      0      0 0.2849175167
## D6M201    6       1 23 66 30      0      0 0.3256771455
## D6M15     6       1 22 67 30      0      0 0.2269119375
## D6M294    6       0 19 70 31      0      0 0.0568882383
## D7M246    7      30 22 42 26      0      0 0.6853827910
## D7M145    7       4 28 58 30      0      0 0.9661049965
## D7M62     7       3 28 61 28      0      0 0.8986715993
## D7M126    7      27 29 37 27      0      0 0.1375358089
## D7M105    7       0 30 56 34      0      0 0.6703200460
## D7M259    7      46 17 41 16      0      0 0.6402183775
## D8M94     8       0 28 58 34      0      0 0.6930406201
## D8M339    8       2 28 55 35      0      0 0.5033645208
## D8M178    8       5 25 58 32      0      0 0.6502263262
## D8M242    8      30 23 42 25      0      0 0.7831394949
## D8M213    8       3 30 52 35      0      0 0.3922337029
## D8M156    8      18 25 49 28      0      0 0.8464817249
## D9M247    9       1 30 59 30      0      0 0.9958071340
## D9M328    9      10 30 49 31      0      0 0.5149752882
## D9M106    9       0 27 58 35      0      0 0.5488116361
## D9M269    9      38 22 41 19      0      0 0.8960526581
```

```
## D9M346    9      22 23 43 32      0      0 0.2098789192
## D9M55     9      30 27 42 21      0      0 0.5488116361
## D9M18     9      30 28 41 21      0      0 0.4065696597
## D10M298  10       1 40 59 20      0      0 0.0345431429
## D10M294  10      30 31 45 14      0      0 0.0403117975
## D10M42_  10       4 30 63 23      0      0 0.4259436255
## D10M10   10      11 28 61 20      0      0 0.2560492882
## D10M233  10       0 29 62 29      0      0 0.9355069850
## D11M78   11      30 23 47 20      0      0 0.8278784881
## D11M20   11       3 23 63 31      0      0 0.4093591491
## D11M242  11       9 22 57 32      0      0 0.3900651738
## D11M356  11      31 18 43 28      0      0 0.3090793301
## D11M327  11       0 28 54 38      0      0 0.2385125539
## D11M333  11       3 27 57 33      0      0 0.7074036474
## D12M105  12      30 31 35 24      0      0 0.0628712266
## D12M46   12      33 34 33 20      0      0 0.0083344619
## D12M34   12       2 37 62 19      0      0 0.0551165588
## D12M5    12       6 38 51 25      0      0 0.1207497463
## D12M99   12      30 29 43 18      0      0 0.2385125539
## D12M150  12       5 33 50 32      0      0 0.3727092993
## D13M59   13       0 34 14  7      0     65 0.7007839983
## D13M88   13       6 33 61 20      0      0 0.1715024212
## D13M21   13      17 35 47 21      0      0 0.1006489555
## D13M39   13      32 33 35 20      0      0 0.0232520115
## D13M167  13      31 32 35 22      0      0 0.0427799650
## D13M99   13       0 48 49 23      0      0 0.0007281525
## D13M233  13      14 41 43 22      0      0 0.0050294088
## D13M106  13       0 45 55 20      0      0 0.0036065631
## D13M147  13       0 45 55 20      0      0 0.0036065631
## D13M226  13      28 27 49 16      0      0 0.2207179658
## D13M290  13       1 42 58 19      0      0 0.0112972801
## D13M151  13       1 38 58 23      0      0 0.1453557012
## D14M14   14      32 20 44 24      0      0 0.8337529181
## D14M115  14       9 27 57 27      0      0 0.9602702338
## D14M265  14       4 30 61 25      0      0 0.6902581264
## D14M266  14      30 19 53 18      0      0 0.2385125539
## D15M226  15       8 27 58 27      0      0 0.9310627797
## D15M100  15      18 27 47 28      0      0 0.7235906755
## D15M209  15       1 31 55 33      0      0 0.6880116006
## D15M144  15      30 24 41 25      0      0 0.6930406201
## D15M68   15       3 33 51 33      0      0 0.3823042729
## D15M239  15       0 32 57 31      0      0 0.8535652128
## D15M241  15       5 35 54 26      0      0 0.3995600132
## D15M34   15      42 23 31 24      0      0 0.1913126738
## D16M154  16      30 28 35 27      0      0 0.1071705988
## D16M4    16       1 33 53 33      0      0 0.4916028846
## D16M139  16      31 26 39 24      0      0 0.4844606343
## D16M86   16      30 24 45 21      0      0 0.9048374180
## D17M260  17      30 22 46 22      0      0 0.9780228725
## D17M66   17      30 19 47 24      0      0 0.6930406201
## D17M88   17      15 22 57 26      0      0 0.5838593069
## D17M129  17       3 27 58 32      0      0 0.8041666799
## D18M94   18      30 30 40 20      0      0 0.1888756028
## D18M58   18       7 32 54 27      0      0 0.7175889212
```

```
## D18M106  18      40 23 40 17      0      0 0.6376281516
## D18M186  18       0 35 53 32      0      0 0.4099718965
## D19M68   19      30 23 45 22      0      0 0.9889503893
## D19M117  19      34 26 43 17      0      0 0.3899017615
## D19M65   19      32 27 43 18      0      0 0.3893869039
## D19M10   19      31 26  0  0      0     63 0.3586266707
## DXM186    X       1 69 50  0      0      0 0.0815562004
## DXM64     X       5 52 63  0      0      0 0.3050069459
```

Calculate Genotype Probabilites

Before performing interval mapping or generating effect plots, R/qtl requires genotype probabilities at positions between observed markers. The `calc.genoprob()` function computes these probabilities along each chromosome, accounting for recombination rates and a small genotyping error probability (`error.prob=0.01`). Setting `step=1` calculates genotype probabilities at 1cM intervals, producing a smoother and more accurate representation of the underlying genetic information needed for QTL detection.

```
# Needed for effect plots and interval mapping
cross <- calc.genoprob(cross, step=1, error.prob=0.01)
```

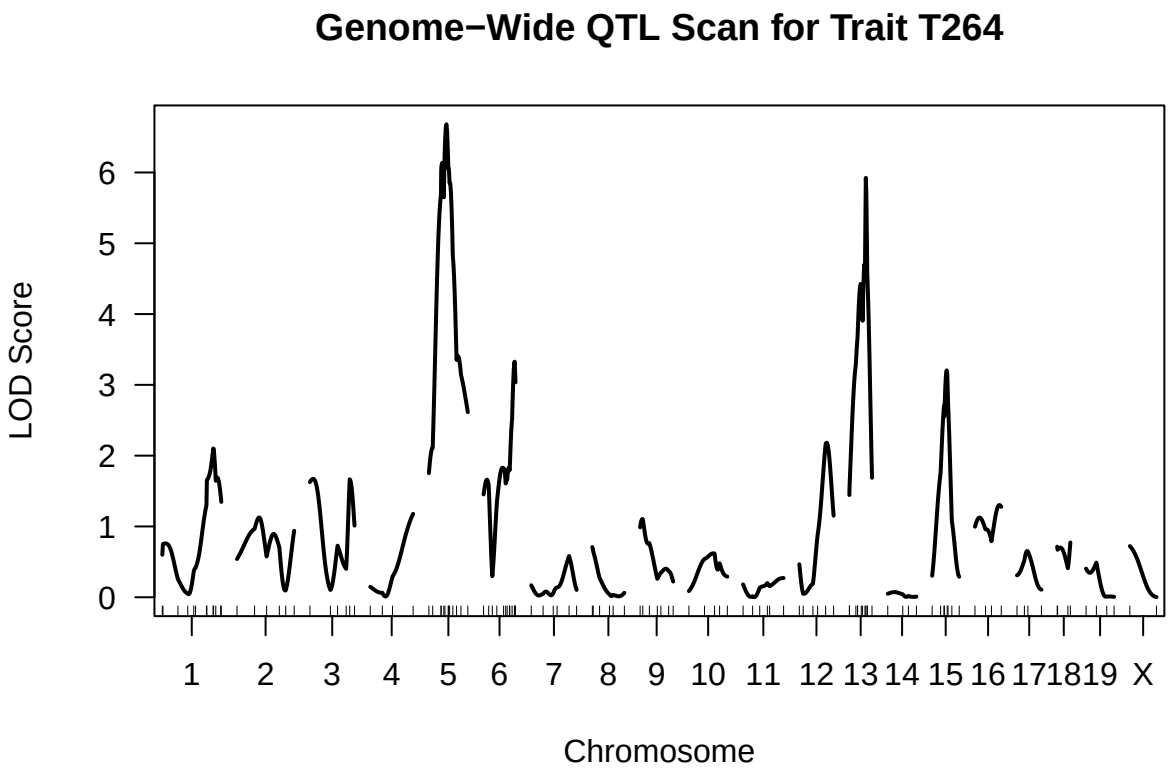
Perform Genome Scan

To identify genomic regions associated with the phenotype, a genome-wide QTL scan is performed using `scanone()`. Here, the analysis uses Haley–Knott regression (`method="hk"`) (Haley & Knott, 1992), a fast and robust approximation for interval mapping. The function tests each genomic position for association with the phenotype `"T264"` and returns a LOD (Logarithm of the Odds) score profile across all chromosomes. The resulting plot visualizes these LOD scores, allowing identification of peaks that may represent putative QTL.

```
# Haley-Knott regression genome scan
scan1 <- scanone(cross, pheno.col="T264", method="hk")
```

```
## Warning in checkcovar(cross, pheno.col, addcovar, intcovar, perm.strata, : Dropping 4 individuals wi
```

```
# Enhanced plot with title and axis labels
plot(scan1,
     main = "Genome-Wide QTL Scan for Trait T264",
     xlab = "Chromosome",
     ylab = "LOD Score",
     col = "black",
     lwd = 2)
```



Permutation Test for Signifcance

To determine whether the observed LOD scores exceed those expected by chance, a permutation test is performed. The function `scanone()` is run 1,000 times with shuffled phenotype labels (`n.perm=1000`), generating an empirical null distribution of LOD scores. This approach provides a robust, data-driven significance threshold that accounts for genome-wide multiple testing. The `summary()` function then compares the observed scan results to the permutation-derived thresholds at the $\alpha = 0.05$ level, identifying which peaks represent statistically significant QTL.

```
perm <- scanone(cross, pheno.col="T264", method="hk", n.perm=1000)
```

```
## Warning in checkcovar(cross, pheno.col, addcovar, intcovar, perm.strata, : Dropping 4 individuals wi
```

```
## Doing permutation in batch mode ...
```

```
summary(scan1, perms=perm, alpha=0.05)
```

```
##          chr  pos  lod
## c5.loc28   5 28.0 6.68
## D13M147   13 26.2 5.92
```

Plotting Significant QTLs

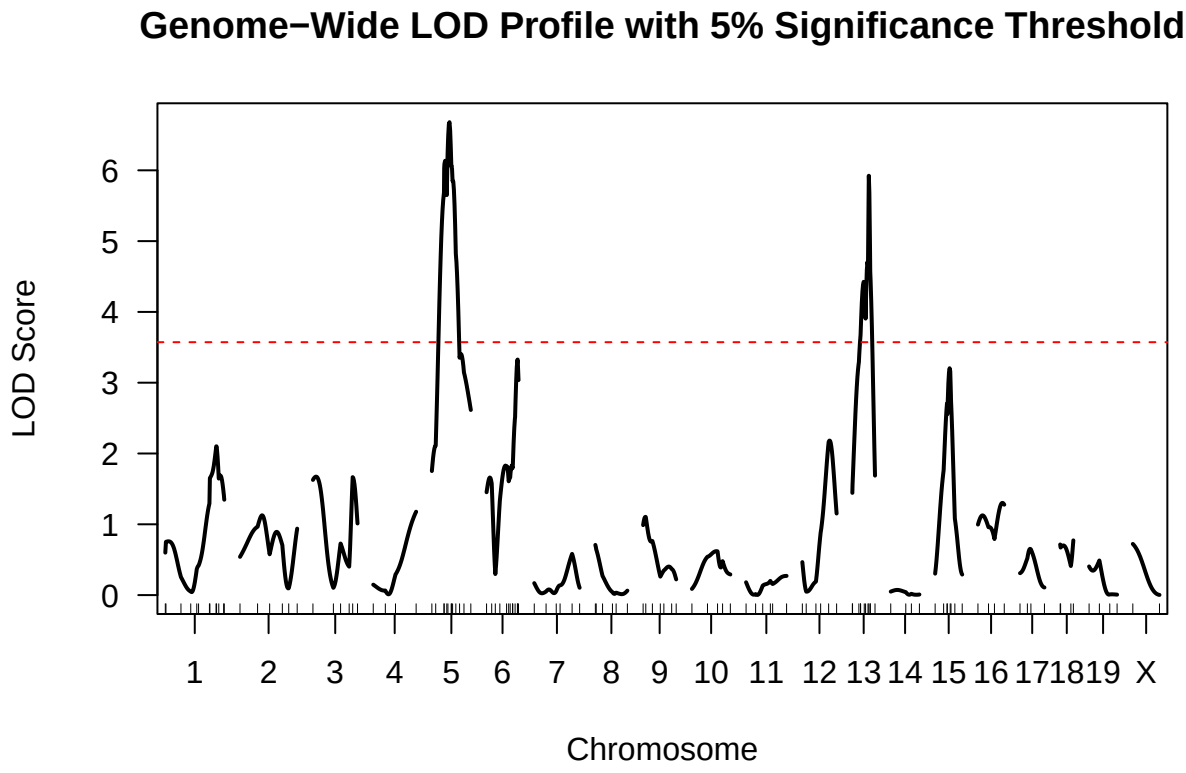
This code visualizes the genome scan results and highlights statistically significant QTLs. First, the 5% genome-wide significance threshold is extracted from the permutation test using `summary(perm, alpha=0.05)`. The LOD score profile from the genome scan is plotted with `plot(scan1)`, and a horizontal dashed red line

(`abline()`) is added at the threshold value. Peaks that rise above this line represent loci that are significant at the genome-wide level.

```
# Get the 5% genome-wide significance threshold
lod_thresh <- summary(perm, alpha=0.05)

# Plot LOD profile with improved labels
plot(scan1,
     main = "Genome-Wide LOD Profile with 5% Significance Threshold",
     xlab = "Chromosome",
     ylab = "LOD Score",
     col = "black",
     lwd = 2)

# Add horizontal line for genome-wide significance
abline(h = lod_thresh, col = "red", lty = 2)
```



Identify QTL Confidence Interval

This code identifies the approximate confidence intervals for detected QTLs on specific chromosomes. The `lodint()` function calculates the region around the peak LOD score where the LOD drops by a specified value (`drop=1.5` corresponds roughly to a 95% confidence interval). In this example, confidence intervals are computed for chromosomes 5 and 13. The resulting intervals indicate the genomic regions most likely to contain the causal loci underlying the trait of interest.

```r
# LOD interval (95% confidence) for a chromosome
ci5 <- lodint(scan1, chr="5", drop=1.5)
ci5
```

```
##          chr pos      lod
## c5.loc15   5  15 5.054036
## c5.loc28   5  28 6.679752
## c5.loc38   5  38 4.842088
```

```r
ci13 <- lodint(scan1, chr="13", drop=1.5)
ci13
```

```
##           chr      pos      lod
## c13.loc17  13 17.00000 4.387866
## D13M147    13 26.15954 5.923396
## c13.loc30  13 30.00000 4.152743
```

Plot Confidence Intervals

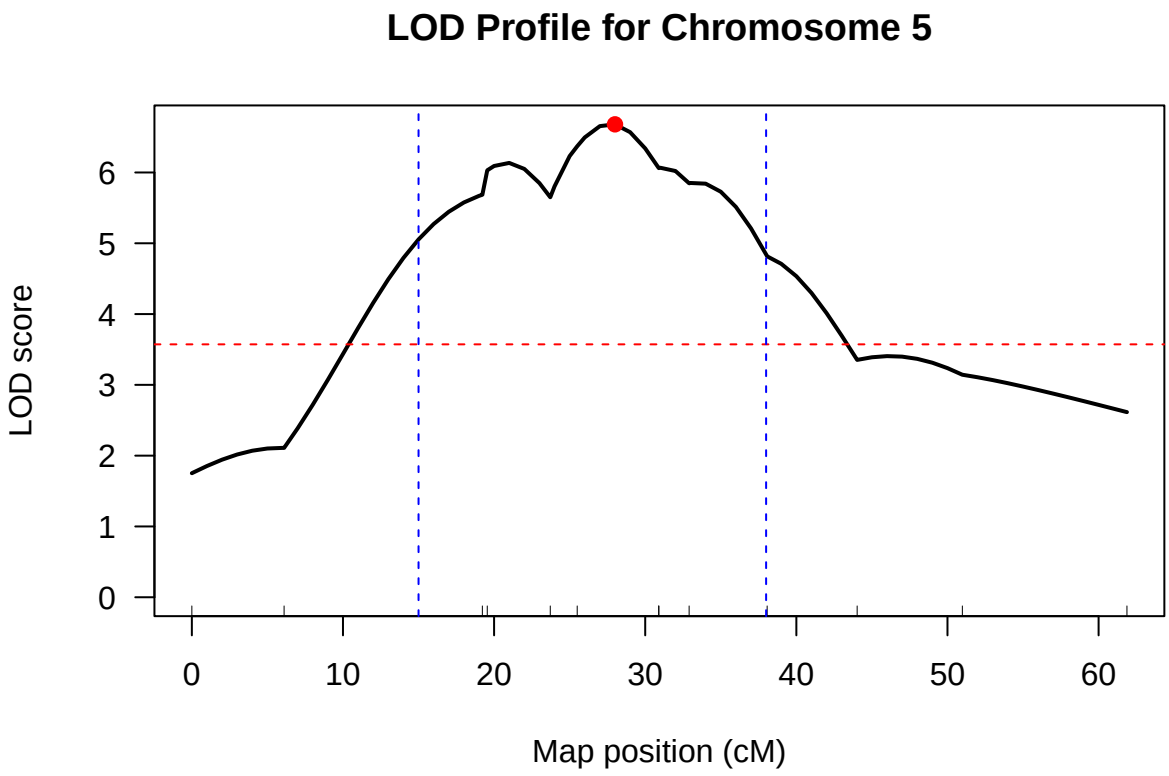
This section visualizes the LOD profiles for chromosomes 5 and 13 from the genome-wide QTL scan. Horizontal red dashed lines represent the 5% genome-wide significance threshold, indicating which peaks are statistically significant. Blue vertical dashed lines denote the 95% confidence intervals around detected QTLs, providing a range where the true QTL is likely located. Red points mark the peak markers within these intervals (`c5.loc28` for chromosome 5 and `D13M147` for chromosome 13), highlighting the loci with the strongest association to the trait of interest. This visualization helps to quickly identify significant QTLs and their approximate genomic locations.

```r
# LOD profile plot for chromosome 5
plot(scan1, chr=5, main="LOD Profile for Chromosome 5", ylab="LOD score")

# Add horizontal significance threshold
abline(h=lod_thresh, col="red", lty=2)

# Add vertical lines for CI
abline(v=15, col="blue", lty=2) # start of CI
abline(v=38, col="blue", lty=2) # end of CI

# Peak marker
points(x=28, y=6.68, col="red", pch=19) # c5.loc28
```

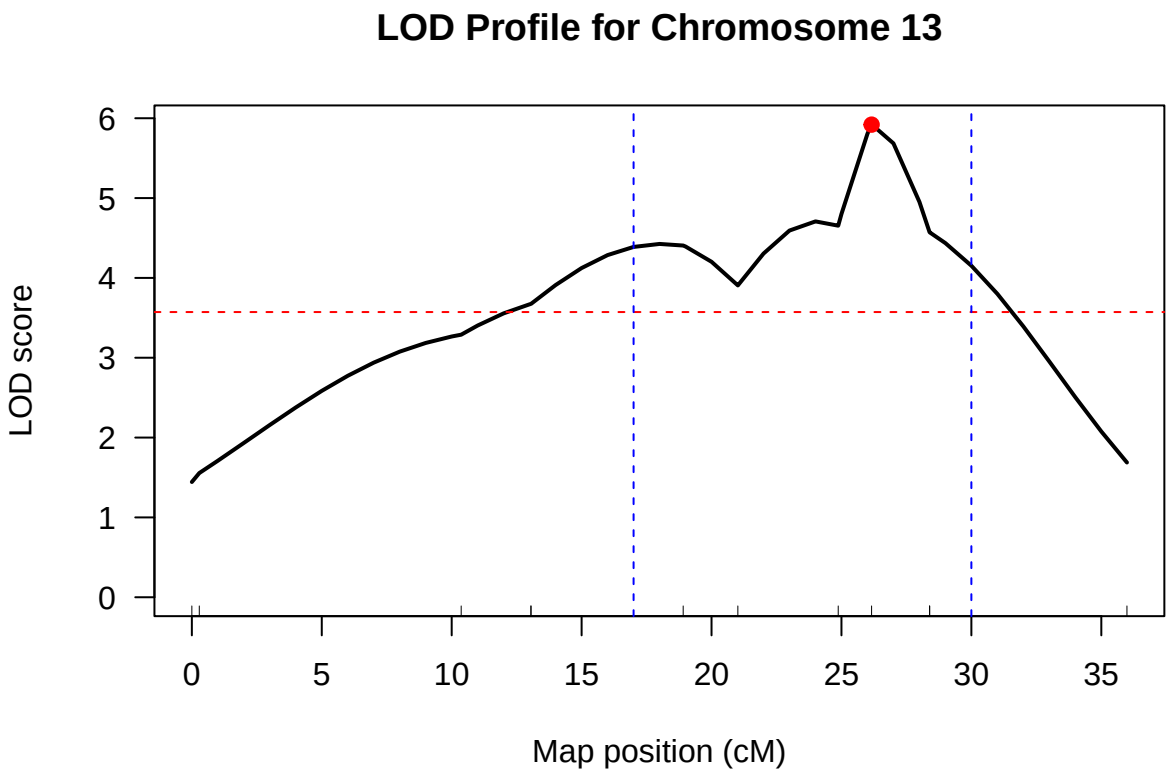



```
# LOD profile plot for chromosome 13
plot(scan1, chr=13, main="LOD Profile for Chromosome 13", ylab="LOD score")

# Add horizontal significance threshold
abline(h=lod_thresh, col="red", lty=2)

# Add vertical lines for CI
abline(v=17, col="blue", lty=2) # start of CI
abline(v=30, col="blue", lty=2) # end of CI

# Peak marker
points(x=26.16, y=5.92, col="red", pch=19) # D13M147
```

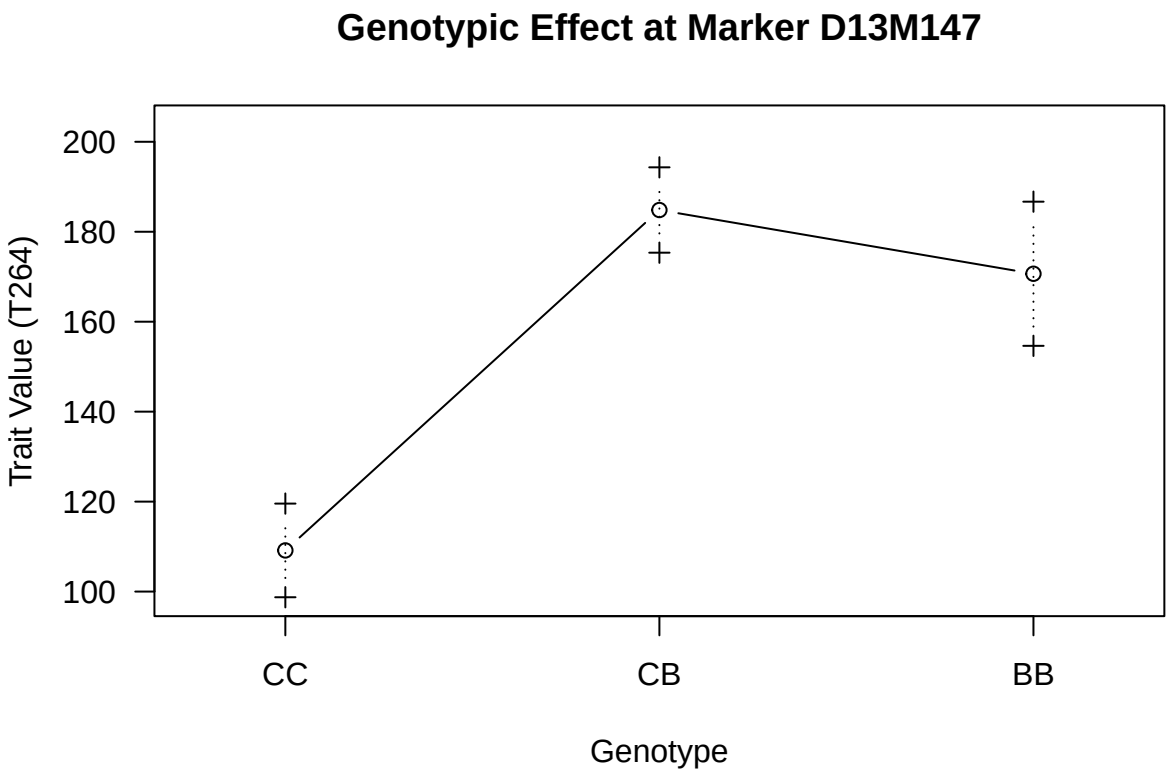


Plot Marker Effects

This code visualizes the effect of a specific marker on the phenotype. The `effectplot()` function shows how different genotypes at the chosen marker (here, `D13M147`) influence the trait values. This allows you to assess the magnitude and direction of the genetic effect at that locus. Note that if genotype probabilities were not previously calculated with `calc.genoprob()`, the function will automatically perform that step before plotting.

```
# Plot effect at marker D13M147 with enhanced visualization
effectplot(cross, mname1 = "D13M147",
           main = "Genotypic Effect at Marker D13M147",
           xlab = "Genotype",
           ylab = "Trait Value (T264)")
```

```
## Warning in effectplot(cross, mname1 = "D13M147", main = "Genotypic Effect at
## Marker D13M147", : -Running sim.geno.
```



Conclusion

This workflow demonstrates a complete pipeline for QTL mapping using R/qtl, from importing genotype and phenotype data to performing genome scans, permutation testing, identifying significant QTLs, and estimating confidence intervals and effect sizes. Using the `listeria` F2 intercross dataset as a proof-of-concept, all steps including data quality checks, genotype probability calculation, and effect plotting were successfully executed, confirming that the pipeline functions as intended. Once genotype and phenotype data from the F2 Poecilia population are available, this workflow can be directly applied, enabling identification of loci associated with dorsal fin traits and facilitating downstream analyses to investigate the genetic architecture of sexually selected traits. The modular structure of this pipeline ensures reproducibility and allows easy adaptation to new datasets, making it a reliable tool for quantitative genetics studies.

References

Broman, K. W., Wu, H., Sen, Ś., & Churchill, G. A. (2003). R/qtl: QTL mapping in experimental crosses. Bioinformatics, 19(7), 889–890. https://doi.org/10.1093/bioinformatics/btg112

Haley, C. S., & Knott, S. A. (1992). A simple regression method for mapping quantitative trait loci in line crosses using flanking markers. Heredity, 69(4), 315–324. https://doi.org/10.1038/hdy.1992.131