Rice Cold Tolerance - QTLseqr Quick-Guide

Norman Warthmann, PBG Laboratory

6/15/2022

Introduction

QTLseqr is an R package for Bulk Segregant Analysis from NGS data. QTLseqr was developed and published by Ben N. Mansfeld and Rebecca Grumet. We forked their github repository (https://github.com/bmansfeld/QTLseqr/) and made minor changes to their software to adopt it to our needs. For detailed instructions on usage and theory of the statistics please consult the vignette of the original QTLseqr package: https://github.com/bmansfeld/QTLseqr/raw/master/vignettes/QTLseqr.pdf.

If you use QTLseqr, please cite:

> Mansfeld B.N. and Grumet R, QTLseqr: An R package for bulk segregant analysis with next-generation sequencing *The Plant Genome* (2018) doi:10.3835/plantgenome2018.01.0006

Example Analysis

The data for below example analysis is drawn from a study on cold tolerance in rice by Yang, Z et al. (2013). It is the same data that Mansfeld and Grumet had used. We only slightly modified their instructions taken from their github page https://github.com/bmansfeld/QTLseqr and vignette.

> Yang Z, Huang D, Tang W, Zheng Y, Liang K, Cutler AJ, et al. (2013) Mapping of Quantitative Trait Loci Underlying Cold Tolerance in Rice Seedlings via High-Throughput Sequencing of Pooled Extremes. PLoS ONE 8(7): e68433. https://doi.org/10.1371/journal.pone.0068433

The raw sequencing data for Yang Z, et al. (2013) is available from NCBI's short read archive in BioProjet PRJNA198759:

- Run SRR834931: Rice ET pool (385 extremely tolerant individuals)
- Run SRR834927: Rice ES pool (430 extremely sensitive individuals)

We have downloaded the raw data (fasterq-dump SRR834931 SRR834927) and performed alignment (bwa) and variant calling (freebayes) against the Nipponbare reference genome (IRGSP-1.0) with our PBGL snakemake workflow (https://github.com/pbgl/dna-proto-workflow). QTLseqr analysis on the resulting VCF file produced by freebayes can be performed as outlined below.

Installation

Install the PBGL version of QTLseqr from github like so:

```r
#install the dependencies
install.packages(c( "data.table",
                    "modeest",
                    "locfit",
                    "dplyr",
                    "tidyr",
                    "vcfR",
                    "ggplot2"), dependencies=TRUE)

# install devtools
install.packages("devtools", dependencies=TRUE)

# use devtools to install QTLseqr from github
library("devtools")
#devtools::install_github("bmansfeld/QTLseqr")
#devtools::install_github("warthmann/QTLseqr")
devtools::install_github("pbgl/QTLseqr")
```

Note: When installing above R-packages, you may run into missing system dependencies. On a clean Ubuntu systems these may include libssl-dev liblapack-dev libopenblas-dev libcurl4-openssl-dev libxml2-dev libudunits2-dev libmariadbclient-dev gfortran libpng-dev libjpeg-dev. Some functions of QTLseqr (e.g., counting SNPs) are implemented in C++. Hence, the install of QTLseqr from github requires the compiler. On Linux this should work out of the box, for Windows and Mac you will need Rtools or Xcode, respectively. You might need assistance from your system administrator.

Do the install from github if you can. Alternatively, we provide binaries of the pbgl version of the QTLseqr R-package for local install: Linux, Mac, Windows. They were compiled with R version 4.2.

```r
#load dependencies
library("devtools")
library("data.table")
library("dplyr")
library("tidyr")
library("vcfR")
library("ggplot2")
library("QTLseqr")
```

Step-by-Step Guide

Define the input:

```
#Set samples
HighBulk <- "ET-pool-385" # SRA-run: SRR834931
LowBulk <- "ES-pool-430" # SRA-run: SRR834927

#set file name of the VCF file to load
file <- "wGQ-Filt-freebayes~bwa~IRGSP-1.0~both-segregant_bulks~filtered-default.vcf"

#Specify which chromosomes should be included in the analysis (i.e., exclude smaller contigs)
Chroms <- c("NC_029256.1",
            "NC_029257.1",
            "NC_029258.1",
            "NC_029259.1",
            "NC_029260.1",
            "NC_029261.1",
            "NC_029262.1",
            "NC_029263.1",
            "NC_029264.1",
            "NC_029265.1",
            "NC_029266.1",
            "NC_029267.1")
```

Chromosome information can be found in the VCF file header. Open the VCF file in a text editor and find the '##contig' lines. E.g.:

```
##reference=genomes_and_annotations/IRGSP-1.0/GCF_001433935.1_IRGSP-1.0_genomic.fna
##contig=<ID=NC_029256.1,length=43270923>
##contig=<ID=NC_029257.1,length=35937250>
##contig=<ID=NC_029258.1,length=36413819>
##contig=<ID=NC_029259.1,length=35502694>
##contig=<ID=NC_029260.1,length=29958434>
##contig=<ID=NC_029261.1,length=31248787>
##contig=<ID=NC_029262.1,length=29697621>
##contig=<ID=NC_029263.1,length=28443022>
##contig=<ID=NC_029264.1,length=23012720>
##contig=<ID=NC_029265.1,length=23207287>
##contig=<ID=NC_029266.1,length=29021106>
##contig=<ID=NC_029267.1,length=27531856>
```

Load the data from the VCF file:

```
df <-
  importFromVCF(
    file = file,
    highBulk = HighBulk,
    lowBulk = LowBulk,
    chromList = Chroms
    )
```

Filter the Variants (user-defined criteria)

```
df_filt <-
    filterSNPs(
        SNPset = df,
        refAlleleFreq = 0.20,
        minTotalDepth = 70,
        maxTotalDepth = 200,
        minSampleDepth = 30,
        depthDifference = 100,
        minGQ = 150,
        verbose = TRUE
    )
```

```
## Filtering by reference allele frequency: 0.2 <= REF_FRQ <= 0.8

## ...Filtered 112443 SNPs

## Filtering by total sample read depth: Total DP >= 70

## ...Filtered 344048 SNPs

## Filtering by total sample read depth: Total DP <= 200

## ...Filtered 335778 SNPs

## Filtering by per sample read depth: DP >= 30

## ...Filtered 25011 SNPs

## Filtering by Genotype Quality: GQ >= 150

## ...Filtered 459528 SNPs

## Filtering by difference between bulks <= 100

## ...Filtered 13 SNPs

## Original SNP number: 1714745, Filtered: 1276821, Remaining: 437924
```

Run the G' (G prime) Analysis

```
df_filt <-
    runGprimeAnalysis(
        SNPset = df_filt,
        windowSize = 1e6,
        outlierFilter = "deltaSNP",
        filterThreshold = 0.1
    )
```

Run the Delta SNP Analysis

```
df_filt <-
    runQTLseqAnalysis(
        SNPset = df_filt,
        windowSize = 1e6,
        popStruc = "F2",
        bulkSize = c(385, 430),
        replications = 10000,
        intervals = c(95, 99)
    )
```

Plot SNP Density

```
#Plot SNP density
plotQTLStats(SNPset = df_filt, var = "nSNPs", plotIntervals = TRUE)
```

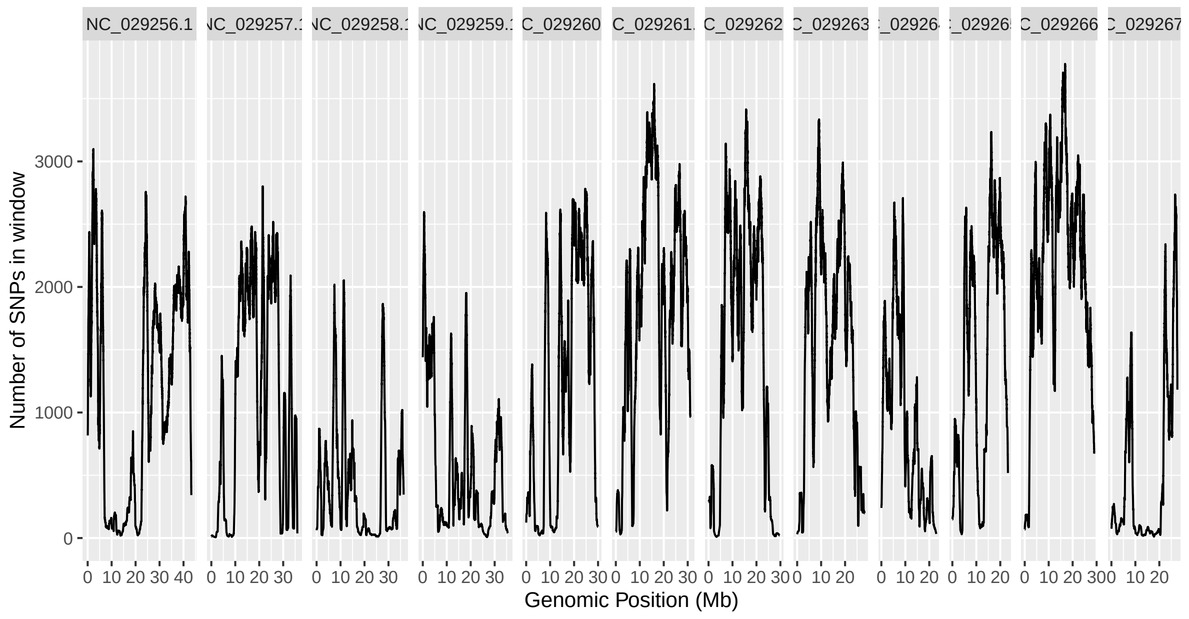

Figure 1: SNP Density

Plot G'

```
plotQTLStats( SNPset = df_filt,
              var = "Gprime",
              plotThreshold = TRUE,
              q = 0.02
              )
```

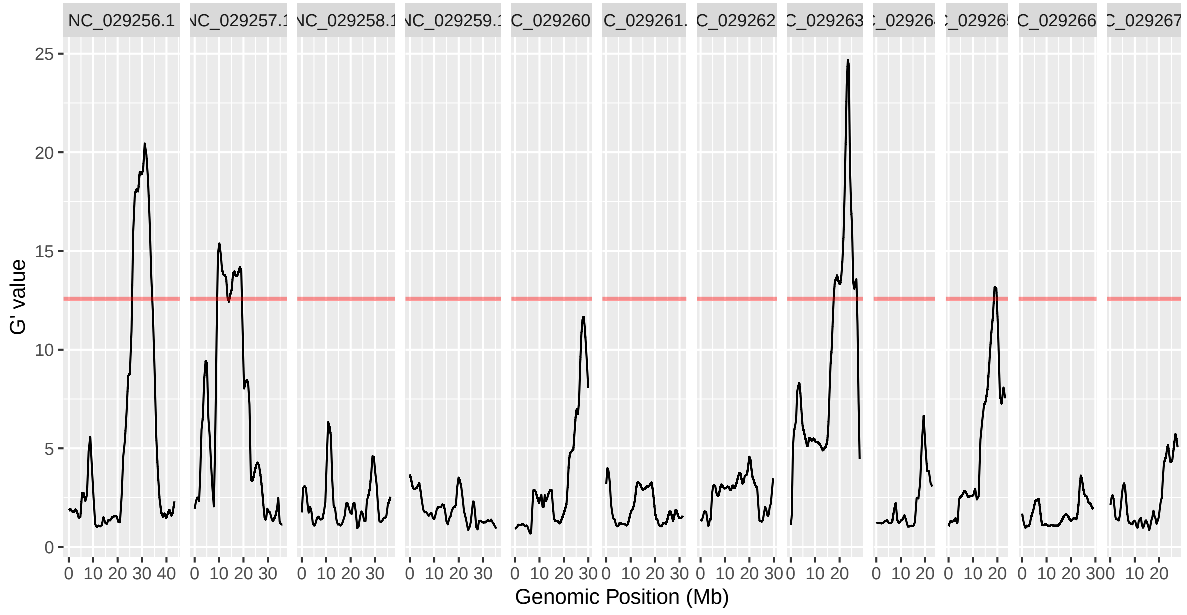

Figure 2: G' Values

```
#get a list of significant Loci
getQTLTable(SNPset = df_filt, method = "Gprime", alpha=0.02)
```

```
##         CHROM qtl    start      end  length nSNPs avgSNPs_Mb peakDeltaSNP
## 1 NC_029256.1   1 25919888 34179239 8259351 10902       1320   -0.3539189
## 2 NC_029257.1   2  9276902 13615518 4338616  6754       1557   -0.3038703
## 3 NC_029257.1   3 14279401 19317487 5038086 10454       2075   -0.2952875
## 4 NC_029263.1   4 17588363 27078814 9490451 14524       1530    0.3885687
## 5 NC_029265.1   5 18569169 19745046 1175877  2901       2467   -0.2860809
##   posPeakDeltaSNP avgDeltaSNP maxGprime posMaxGprime meanGprime  sdGprime
## 1        31099041  -0.3305772  20.45147     31099041   17.99497 1.7329662
## 2        10113941  -0.2919851  15.38594     10113941   13.94807 0.7187341
## 3        18531185  -0.2923448  14.18050     18531795   13.65530 0.4505057
## 4        23374729   0.3094109  24.66993     23374729   15.77846 3.5120489
## 5        19573118  -0.2837809  13.17250     18849135   13.02287 0.1729573
##        AUCaT     meanPval   meanQval
## 1 43433062.4 0.0004382754 0.01504953
## 2  6640787.3 0.0013275507 0.01869950
## 3  5390516.6 0.0014415890 0.01872612
## 4 34282532.7 0.0010584604 0.01748016
## 5   533491.5 0.0017780554 0.01886546
```

Plot Delta SNP

```
#Plot Delta-SNP
plotQTLStats(SNPset = df_filt,
             var = "deltaSNP",
             plotIntervals = TRUE)
```

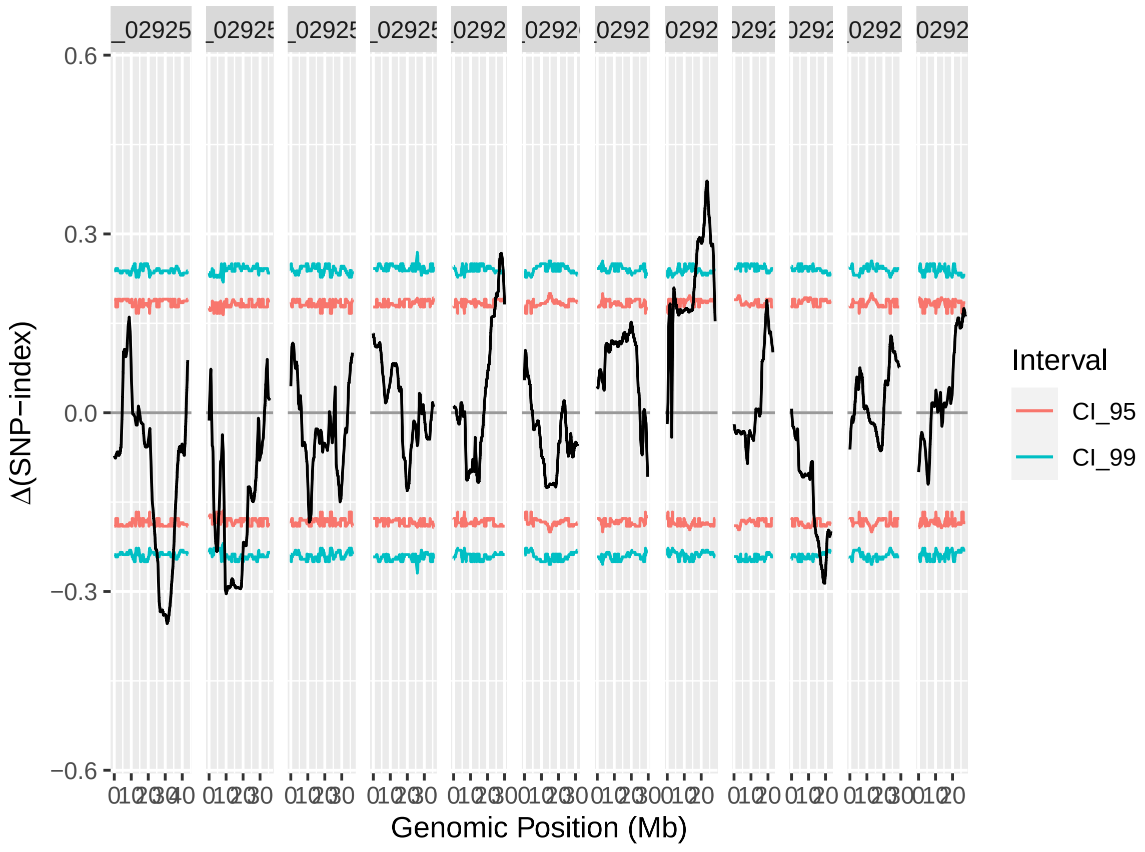

Figure 3: Delta SNP

```
#export table as csv file
getQTLTable(SNPset = df_filt, method = "QTLseq",
            export=TRUE, fileName= "my_first_BSA_result.csv")
```

```
##          CHROM qtl    start      end   length nSNPs avgSNPs_Mb peakDeltaSNP
## 1 NC_029256.1   1 25374097 34909843  9535746 12364       1297   -0.3539189
## 2 NC_029257.1   2  4415438  5091602   676164  1125       1664   -0.2327032
## 3 NC_029257.1   3  9332841 19768259 10435418 18240       1748   -0.3038703
## 4 NC_029260.1   4 26756627 29113403  2356776  3949       1676    0.2673995
## 5 NC_029263.1   5 16699338 27451398 10752060 16576       1542    0.3885687
## 6 NC_029265.1   6 16868410 20552240  3683830  8886       2412   -0.2860809
##   posPeakDeltaSNP avgDeltaSNP
## 1        31099041  -0.3218026
## 2         4500597  -0.2319840
## 3        10113941  -0.2914959
## 4        28084978   0.2596036
## 5        23374729   0.3025358
## 6        19573118  -0.2678312
```

Plot only a subset of Chromosomes

```
#Plot only a subset of Chromosomes
plotQTLStats( SNPset = df_filt,
              var = "Gprime",
              plotThreshold = TRUE,
              q = 0.02,
              subset=c("NC_029256.1","NC_029257.1", "NC_029260.1", "NC_029263.1")
)
```

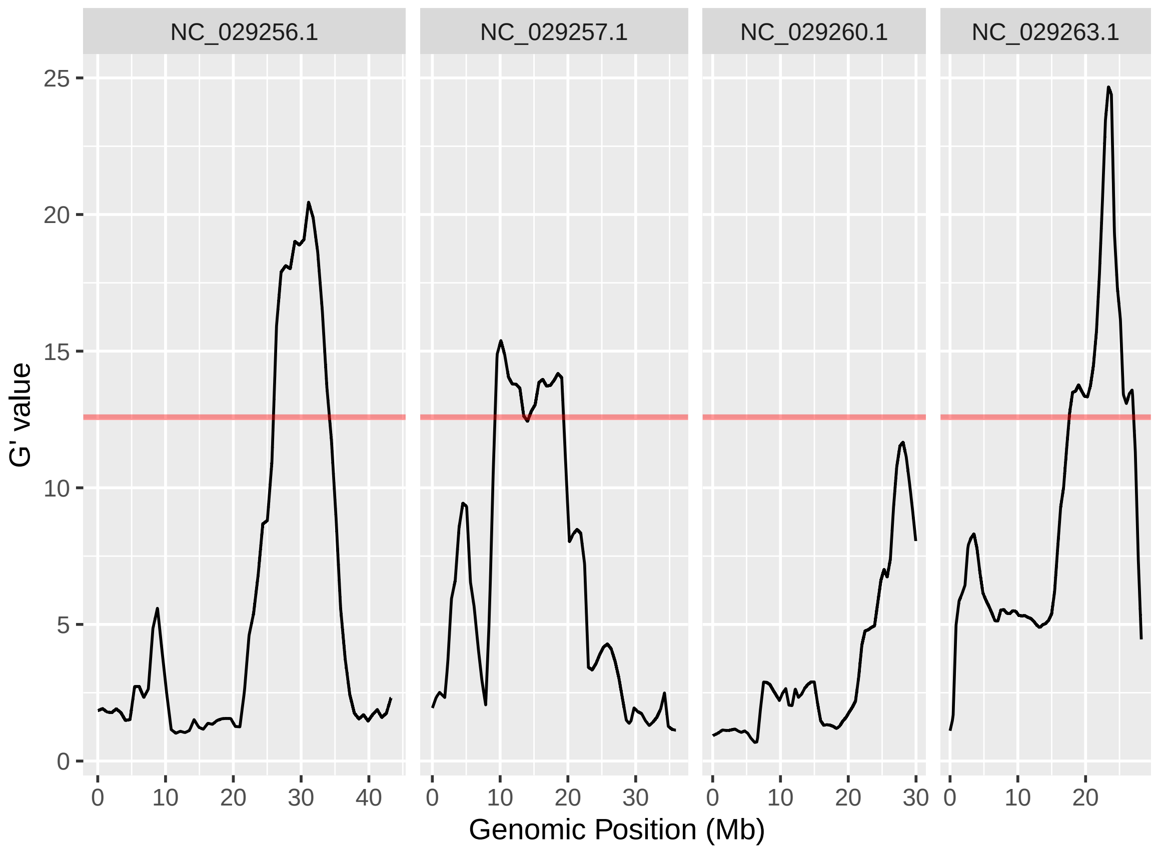

Figure 4: G' Chromsome 1,2,5,8

```r
#Plot only a subset of Chromosomes
plotQTLStats( SNPset = df_filt,
              var = "deltaSNP",
              plotIntervals = TRUE,
              subset=c("NC_029256.1","NC_029257.1", "NC_029260.1", "NC_029263.1")
)
```

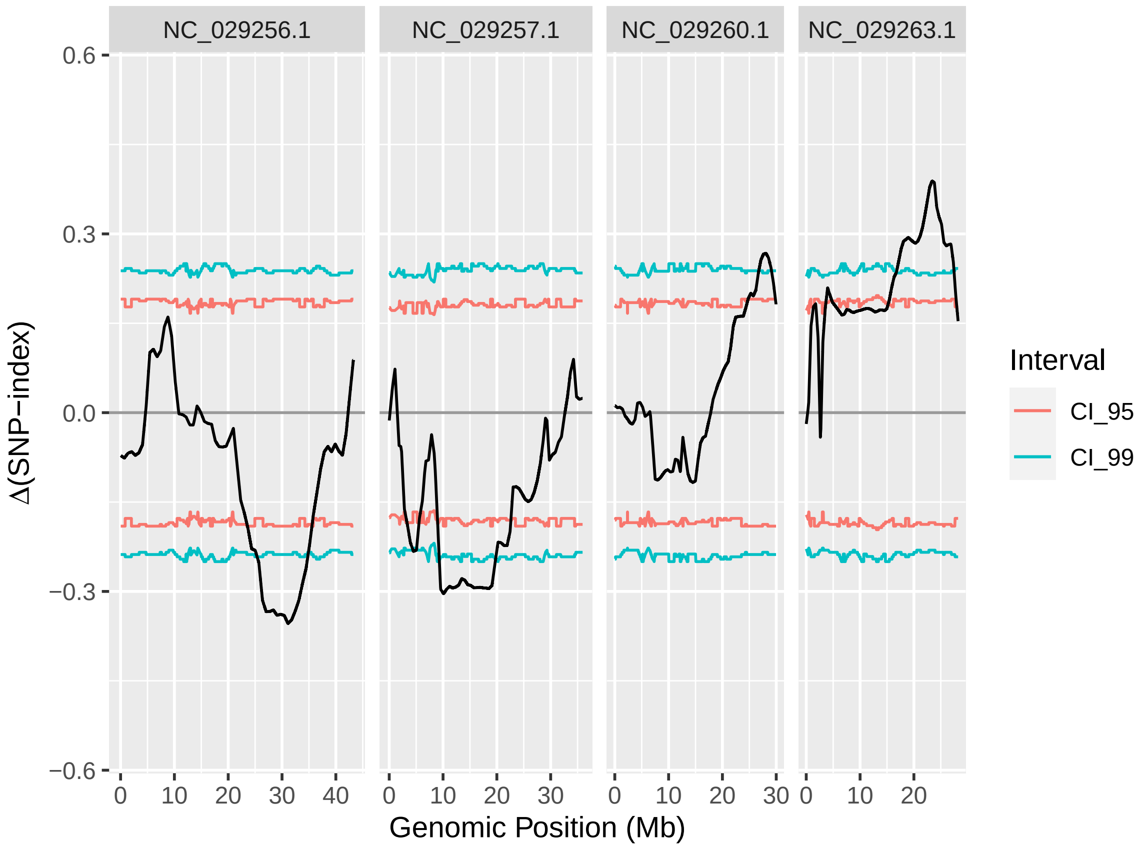

Figure 5: G' Chromsome 1,2,5,8