GWASpoly Version 2

Jeff Endelman

Overview

The following table outlines the basic workflow:

Function	Purpose
read.GWASpoly	Read in the marker and phenotype data
set.K	Specify covariance matrix for the polygenic effect
set.params	Specify additional model parameters
GWASpoly	Perform the mixed model association test
qq.plot	Check inflation of p-values
set.threshold	Establish QTL detection threshold
manhattan.plot	Visualize p-value results
get.QTL	Extract significant markers
fit.QTL	Compute partial R2 for multi-QTL model
write.GWASpoly	Write all results to file (optional)

Datasets

Two different datasets are distributed with the package. The original marker (TableS1.csv) and phenotype (TableS2.csv) files used in this vignette were published as supplemental files by [Rosyara et al. (2016)] ((https://doi.org/10.3835/plantgenome2015.08.0073). In June 2021, as part of the development of new software features, the vignette was updated to use a larger dataset from the University of Wisconsin potato breeding program.

Read in data

The `system.file` command can be used to obtain the path to these files, which get installed in the subdirectory "extdata." You will not need to do this when analyzing your own data; just pass the names of the files as strings.

```
genofile <- system.file("extdata", "new_potato_geno.csv", package = "GWASpoly")
phenofile <- system.file("extdata", "new_potato_pheno.csv", package = "GWASpoly")

library(GWASpoly)
data <- read.GWASpoly(ploidy=4, pheno.file=phenofile, geno.file=genofile,
                      format="numeric", n.traits=1, delim=",")
```

```
## Number of polymorphic markers: 9888
## N = 957 individuals with phenotypic and genotypic information
## Detected following fixed effects:
## env
## Detected following traits:
## vine.maturity
```

The first three columns of the geno.file contain the marker name, chromosome, and map position (in either cM or bp). Optionally, columns named REF and ALT can be included after the map. Subsequent columns contain the marker data, which can be coded using one of three formats: “ACGT” (i.e., nucleotide), “numeric” (based on dosage of the alternate allele = 0,1,2,..ploidy), and “AB”. Missing marker data are imputed with the population mode (most frequent genotype), but this method is not recommended if there is a lot of missing data (you should use some other software for imputation before using GWASpoly).

The first column of the pheno.file contains the genotype identifier (i.e., the name of the individual or clone), followed by columns of trait data (specify the number of traits with `n.traits`). After the traits can be columns with potential fixed effects. In this example, there is a single trait, potato vine maturity, measured in 6 different environments (“env”).

VCF data

The `VCF2dosage` function can be used to generate the genotype input file for GWASpoly from marker data in Variant Call Format (VCF). The syntax of the function is shown below:

```
VCF2dosage(VCF.file, dosage.file, geno.code, ploidy, samples=NULL,
           min.DP=1, max.missing, min.minor=5)
```

Genotypes from the “GT” (posterior mode) or “DS” (posterior mean) field can be extracted from the VCF file. For more information, consult the reference manual.

Population structure

```
data.loco <- set.K(data,LOCO=TRUE,n.core=2)
data.original <- set.K(data,LOCO=FALSE,n.core=2)
```

GWASpoly uses a random polygenic effect to control for population structure, which is called the K model in the GWAS literature (Yu et al. 2006). In the original implementation, all markers were used to calculate a single covariance matrix proportional to MM', where M is the n x m centered genotype matrix for n individuals and m markers. However, because this polygenic model is equivalent to including all of the markers as random effects, when each marker is tested as a fixed effect to calculate the p-value for QTL discovery, it is also present as a random effect. This overlap, called “proximal contamination” by Listgarten et al. (2012), slightly reduces model performance. For computational efficiency, Yang et al. (2014) proposed the leave-one-chromosome-out (LOCO) method, in which a different covariance matrix is calculated for each chromosome based on the markers from all other chromosomes. To use this method in GWASpoly, use the `LOCO=TRUE` option in the function `set.K`. Multiple cores can be used to speed up the calculation with the argument `n.core`.

In many situations, the K model is sufficient to control population structure. The main diagnostic to assess this is a QQ plot (see below). If additional control is desired, the function `set.params` can be used to specify the inclusion of principal components as fixed effects (P+K model), or the user can supply their own population covariates in the phenotype input file (Q+K model).

Marker curation

GWAS relies on linkage disequilibrium (LD) between markers and QTL for detection. Even for markers near a QTL, differences in allele frequency at the two loci reduce LD, which is one reason GWAS p-values are not a simple proxy for marker-QTL distance (in contrast to linkage mapping). It follows that markers with low-allele frequency are primarily useful for detecting QTL with similar frequency, but the power to detect rare variants is low.

As a result, it is prudent to remove markers below a frequency threshold using the function `set.params`. Although it is common to do this based on minor allele frequency, a more appropriate metric for heterozygous panels (especially when dominance models are used) is maximum genotype frequency. This threshold can be set using `set.params`, which is also used to specify a fixed effect for “env” of type “factor” (the other option for fixed effects is “numeric”). Based on power calculations, I recommend using 1 - 5/N for the maximum geno.freq. Markers below this threshold have too low power and unnecessarily increase the detection threshold for multiple testing.

```
N <- 957 #Population size
params <- set.params(geno.freq = 1 - 5/N, fixed = "env", fixed.type = "factor")
```

Testing markers for significance

The function `GWASpoly` (same name as the package) conducts the hypothesis tests for each marker. The traditional GWAS model assumes an additive relationship between trait and marker allele dosage. GWASpoly can also test non-additive models. The assumption of complete dominance is denoted “1-dom” because 1 copy of the dominant allele is sufficient. (Consult the manual and Rosyara et al. 2016 for other non-additive models.) Remember that testing more models also increases the probability of a Type I error (unless the threshold is adjusted to compensate). Let’s see how the LOCO and original methods compare for the additive and 1-dom models.

```
data.loco.scan <- GWASpoly(data=data.loco,models=c("additive","1-dom"),
                           traits=c("vine.maturity"),params=params,n.core=2)
```

```
## Analyzing trait: vine.maturity
## P3D approach: Estimating variance components...Completed
## Testing markers for model: additive
## Testing markers for model: 1-dom-alt
## Testing markers for model: 1-dom-ref
```

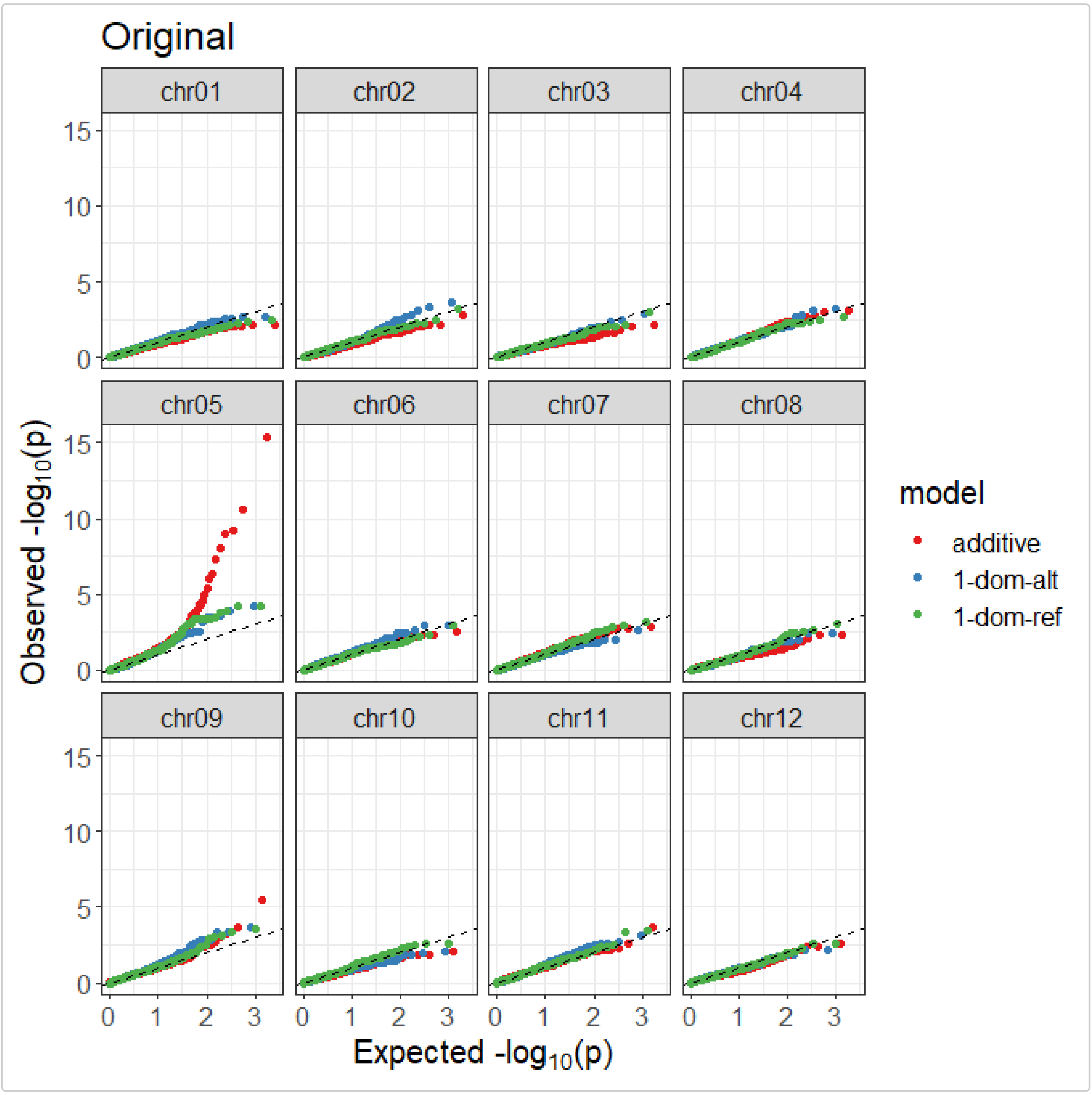
```
data.original.scan <- GWASpoly(data.original,models=c("additive","1-dom"),
                               traits=c("vine.maturity"),params=params,n.core=2)
```

```
## Analyzing trait: vine.maturity
## P3D approach: Estimating variance components...Completed
## Testing markers for model: additive
## Testing markers for model: 1-dom-alt
## Testing markers for model: 1-dom-ref
```

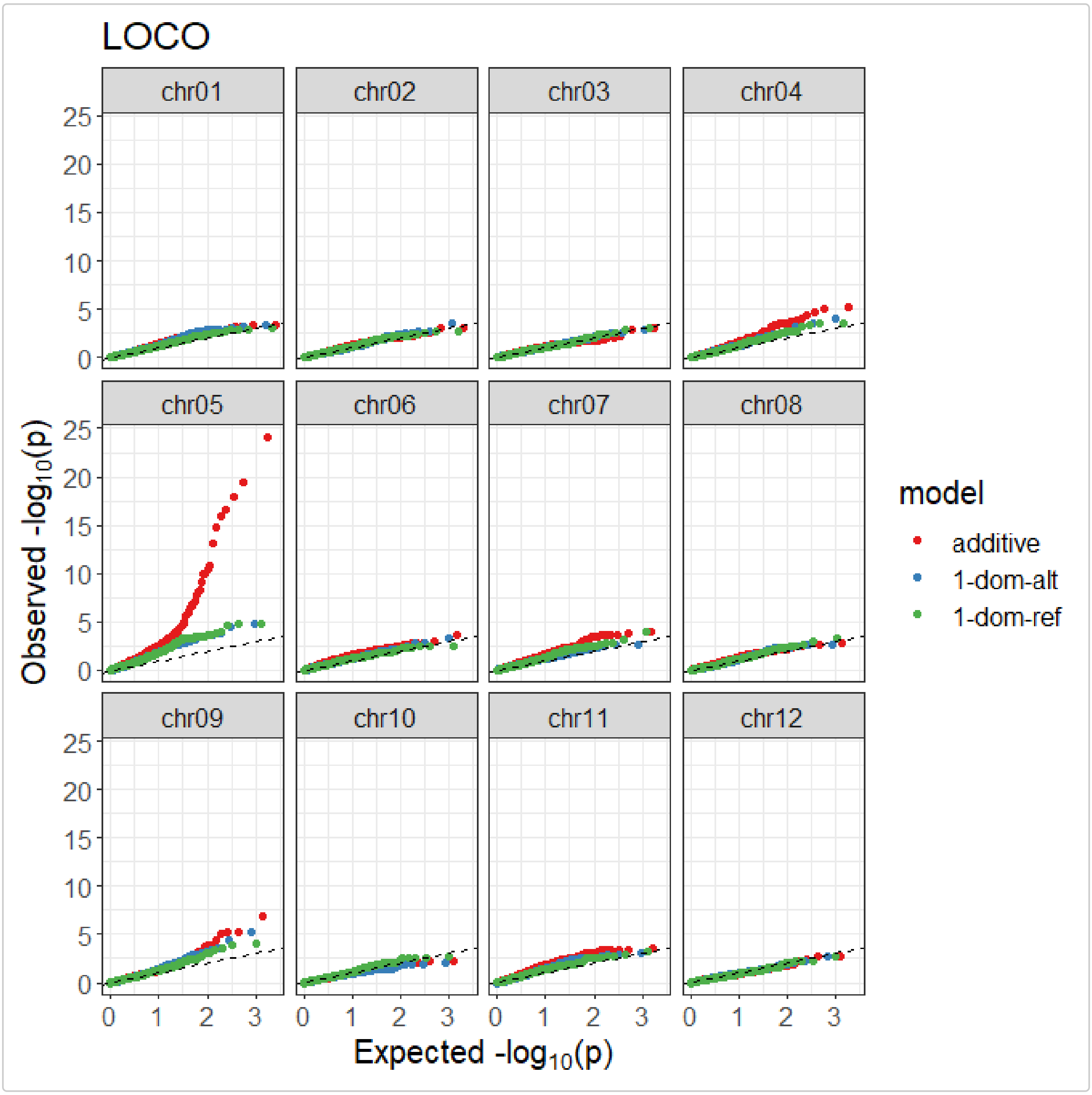
For the 1-dom model, two different analyses are conducted, corresponding to whether the reference or alternate allele is dominant.

One of the standard diagnostics in GWAS is to check the inflation of the -log10(p) values (aka “scores”). This can be done using a quantile-quantile plot of the observed vs. expected values under the null hypothesis, which follows a uniform distribution and is shown with a dotted line. To accommodate the LOCO model, results are shown for each chromosome.

```
library(ggplot2)
qq.plot(data.original.scan,trait="vine.maturity") + ggtitle(label="Original")
```



```
qq.plot(data.loco.scan,trait="vine.maturity") + ggtitle(label="LOCO")
```



The association scores were higher under LOCO, which is to be expected because no other markers on the chromosome are present under the null hypothesis. The LOCO method is also advantageous for estimating partial R2 values with multiple QTL (see below).

There are several methods for establishing a -log10(p) threshold with `set.threshold`, to control the genome-wide false positive rate. For the original "Bonferroni" method, the overall significance level is divided by the number of tested markers. However, using the total number of tests is overly conservative because the tests are not independent due to LD. The LD between markers can be used to estimate an effective number of markers using method="M.eff". One can also use a permutation test (method="permute"), but this is more computationally intensive. The following code compares the original Boneferroni and M.eff methods:

```
data2 <- set.threshold(data.loco.scan,method="Bonferroni",level=0.05)
```

```
## Thresholds
##               additive 1-dom-alt 1-dom-ref
## vine.maturity      5.3      5.07       5.2
```

```
data2 <- set.threshold(data.loco.scan,method="M.eff",level=0.05)
```

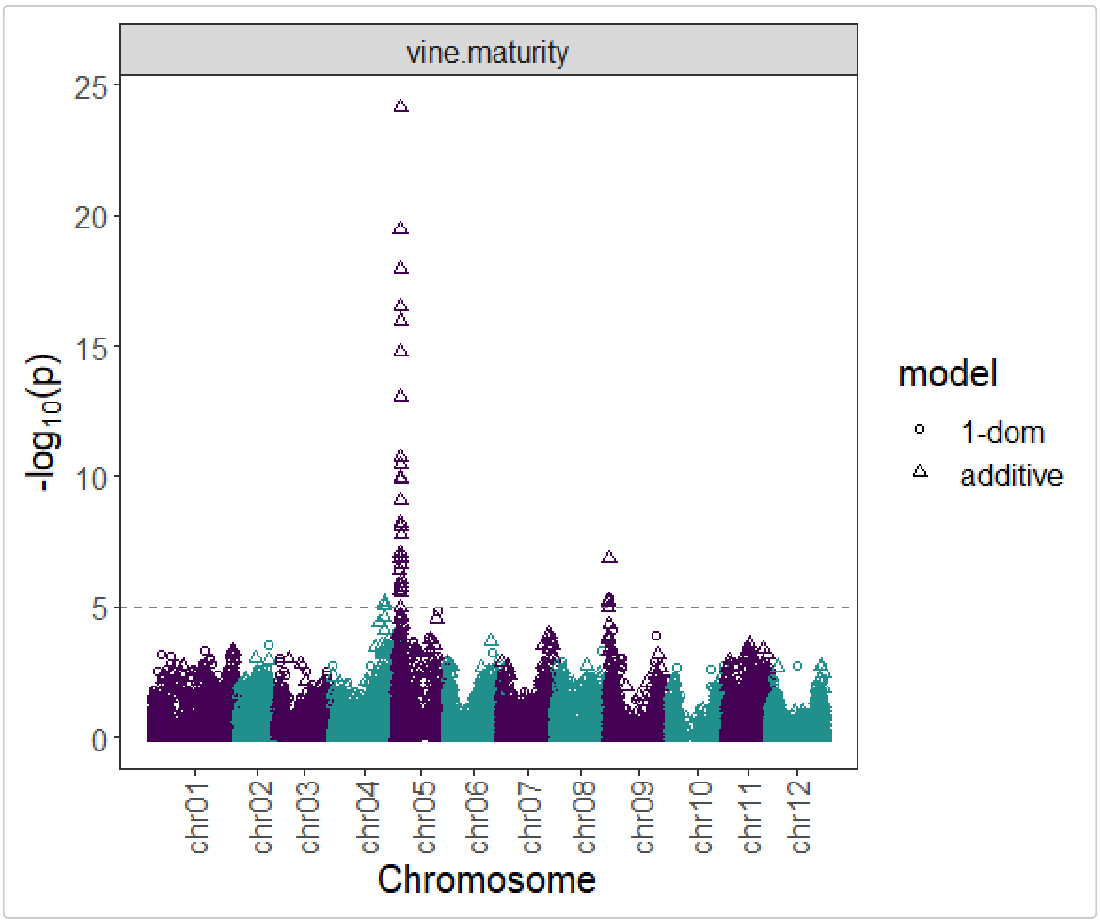
```
## Warning: as(<matrix>, "dspMatrix") is deprecated since Matrix 1.5-0; do
## as(as(as(., "dMatrix"), "symmetricMatrix"), "packedMatrix") instead
```

```
## Thresholds
##               additive 1-dom-alt 1-dom-ref
## vine.maturity        5       4.8      4.91
```

The thresholds for the dominance models are slightly lower than the additive model because, after applying the dominance function to the marker data, some markers are above the threshold for maximum genotype frequency.

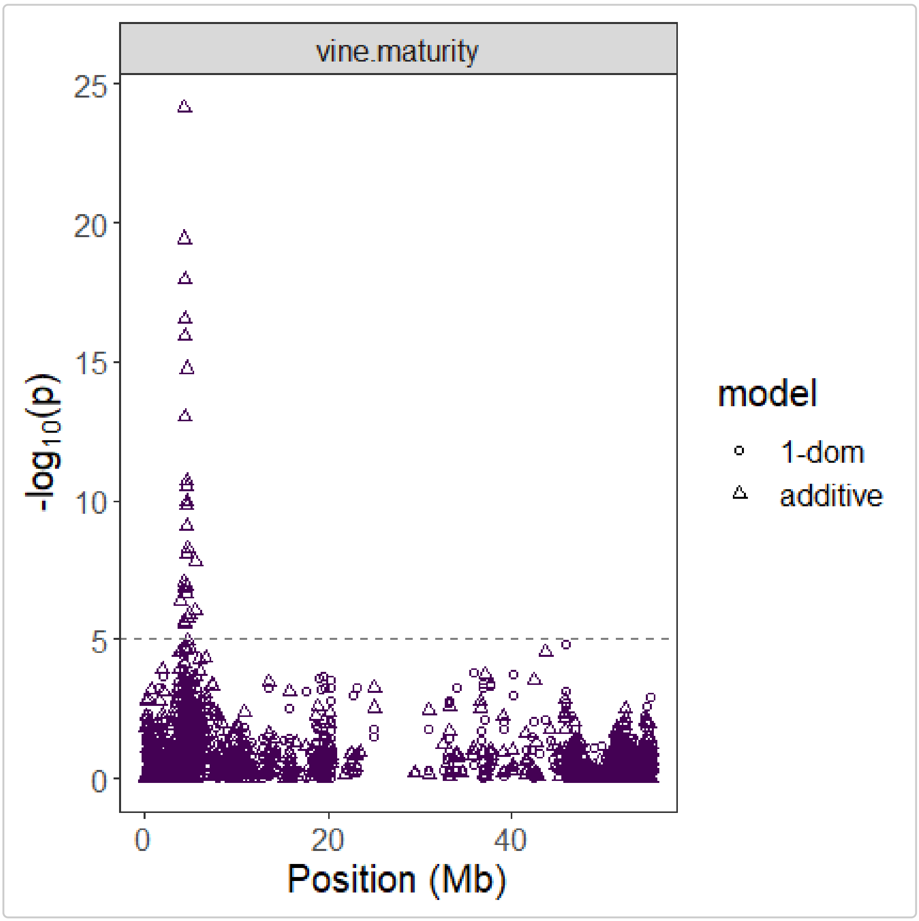
The distribution of -log10(p) values along the genome is visualized using `manhattan.plot`. Because the threshold varies across models, when several models are combined in a single panel, the most stringent value is shown. (To avoid this behavior, make separate figures for each model.) Most of the significant markers were detected with the additive model, with the largest peak on chr05 and smaller peaks on chr04 and chr09.

```
p <- manhattan.plot(data2,traits="vine.maturity")
p + theme(axis.text.x = element_text(angle=90,vjust=0.5))
```



The `manhattan.plot` function can also be used to zoom in on a single chromosome, in this case chr05, which reveals that the peak is located near the CDF1 gene, an important regulator of maturity in potato (Kloosterman et al. 2013)

```
manhattan.plot(data2,traits="vine.maturity",chrom="chr05")
```

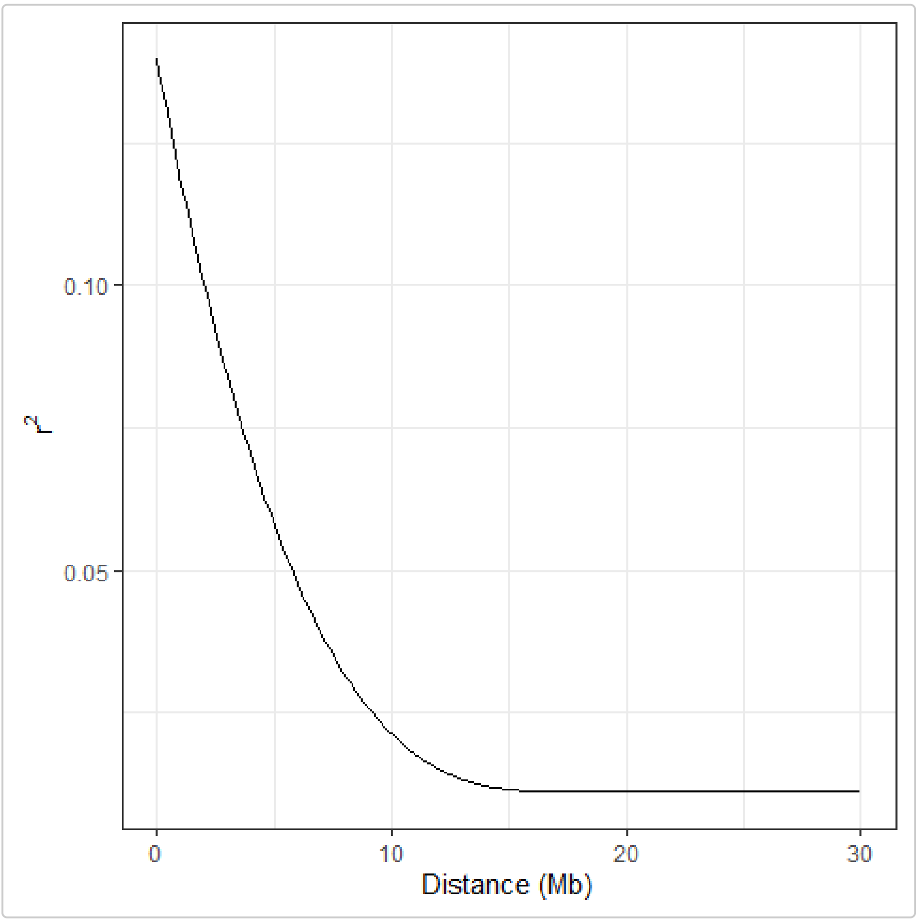


QTL Effects and R2

The function `get.QTL` originally returned all markers with scores above the threshold, but as of June 2021 there is the option to filter the output so that only the most significant marker within a specified window is returned. This is useful for selecting a single marker per QTL. The appropriate window size will depend on the extent of LD in the panel, which can be visualized using the function `LD.plot`.

```
p <- LD.plot(data2, max.loci=1000)
p + xlim(0,30)
```

```
## Warning: Removed 329 rows containing missing values (`geom_line()`).
```



Due to the large number of haplotypes in elite potato germplasm, and because many of the potato SNP array markers are not haplotype-specific, the LD is quite low even at zero distance (Vos et al. 2017). However, from the shape of the curve, a 5-10 Mb window seems appropriate. The argument `max.loci` controls how many markers are used per chromosome for calculating LD. To use all of the markers, set max.loci=NULL.

```r
qtl <- get.QTL(data=data2,traits="vine.maturity",models="additive",bp.window=5e6)
knitr::kable(qtl)
```

	Trait	Model	Threshold	Marker	Chrom	Position	Ref	Alt	Score	Effect
3667	vine.maturity	additive	5	solcap_snp_c2_54335	chr04	56277376	0	1	5.23	-0.2297511
4346	vine.maturity	additive	5	solcap_snp_c2_22964	chr05	4422417	0	1	24.13	-0.6280042
7213	vine.maturity	additive	5	solcap_snp_c2_48597	chr09	793042	0	1	6.89	0.3632297

The function `fit.QTL` can be used to build a multiple QTL model and estimate the partial R^2 for each QTL based on backward elimination. The argument “qtl” is a data frame with two columns: Marker and Model. We also need to include the fixed effect for “env”.

```r
fit.ans <- fit.QTL(data=data2,trait="vine.maturity",
                   qtl=qtl[,c("Marker","Model")],
                   fixed=data.frame(Effect="env",Type="factor"))
knitr::kable(fit.ans,digits=3)
```

	Marker	Chrom	Position	Model	R2	pval
3667	solcap_snp_c2_54335	chr04	56277376	additive	0.019	0
4346	solcap_snp_c2_22964	chr05	4422417	additive	0.079	0
7213	solcap_snp_c2_48597	chr09	793042	additive	0.022	0

The QTL on chr05 explains 8% of the variation vs. 2% for the QTL on chr04 and chr09.