

FQ2

2025-11-07

```
library(lme4)

## Loading required package: Matrix

library(tibble)
library(dplyr)

## 
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
## 
##     filter, lag

## The following objects are masked from 'package:base':
## 
##     intersect, setdiff, setequal, union

library(tidyverse)

## -- Attaching core tidyverse packages ----- tidyverse 2.0.0 --
## vforcats    1.0.0    vreadr      2.1.5
## vggplot2    3.5.2    vstringr   1.5.1
## vlubridate  1.9.4    vtidyrr    1.3.1
## vpurrr      1.0.4

## -- Conflicts ----- tidyverse_conflicts() --
## xtidyr::expand() masks Matrix::expand()
## xdplyr::filter() masks stats::filter()
## xdplyr::lag()    masks stats::lag()
## xtidyr::pack()  masks Matrix::pack()
## xtidyr::unpack() masks Matrix::unpack()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

library(ggthemes)
library(ggrepel)
library(lfmm)
library(RSpectra)
load("pl_lt_t.lmer.RData")
load("growth_light_time.lmer.RData")
source("manhattan_plot.R")
```

```

## 
## Attaching package: 'scales'
## 
## The following object is masked from 'package:purrr':
## 
##     discard
## 
## The following object is masked from 'package:readr':
## 
##     col_factor

#file_path <- file.choose()
#rmarkdown::render(file_path)

```

manhattan plot for 2022

```

pheno_22 <- ranef(pl_lt_t.lmer)$pop %>%
  as_tibble(rownames = "pop", .name_repair = "unique") %>%
  rename(blup_intercept = `Intercept`..., 1,
         blup_light = `mean_light_ly_day2`) %>%
  mutate(model = "Y2022")

## New names:
## * 'Intercept' -> '(Intercept)...1'
## * 'Intercept' -> '(Intercept)...3'

geno_22 <- read.delim("Data/merged_maf_common_UCD2022.tsv", header = TRUE)

Y <- as.matrix(geno_22[,-1])
Y <- t(Y)
X <- matrix(scale(pheno_22$blup_light), ncol = 1)
rownames(X) <- pheno_22$pop
colnames(X) <- "blup_light"
common <- intersect(rownames(X), rownames(Y))
Y <- Y[common,]
X <- X[common, , drop = FALSE]
mod.lfmm <- lfmm_ridge(Y = Y,
                        X = X,
                        K = 2)

pv <- lfmm_test(Y = Y,
                  X = X,
                  lfmm = mod.lfmm,
                  calibrate = "gif")
pvalues <- pv$calibrated.pvalue

plot_data = tibble(loci = geno_22[,1], p_val = pv$pvalue[,1], calibrated.pvalue = pv$calibrated.pvalue[1],
separate_wider_delim(loci, delim = ":" , names = c("chr", "bp")) |>
filter(str_detect(chr, "Chr")) |>
mutate(p.bonf = p.adjust(p_val, method = "bonferroni"),
bp = as.numeric(bp))

```

```

manhattan_22 = plot_manhattan(plot_data,
                             chr_col = "chr",
                             pos_col = "bp",
                             value_col = "calibrated.pvalue",
                             transform = "neglog10",
                             title = "Light association",
                             highlight_top_n = 20)

```

import X and Y data in WL2_2023

```

pheno_23 <- ranef(growth_light_time.lmer)$parent_pop %>%
  as_tibble(rownames = "parent_pop", .name_repair = "unique") %>%
  rename(blup_intercept = `"(Intercept)...1`,
         blup_light = `weekly_avg_SlrW2`) %>%
  mutate(model = "Y2023")

```

```

## New names:
## * '(Intercept)' -> '(Intercept)...1'
## * '(Intercept)' -> '(Intercept)...3'

```

```

geno_23 <- read.delim("Data/merged_maf_common_WL2_2023.tsv", header = TRUE)

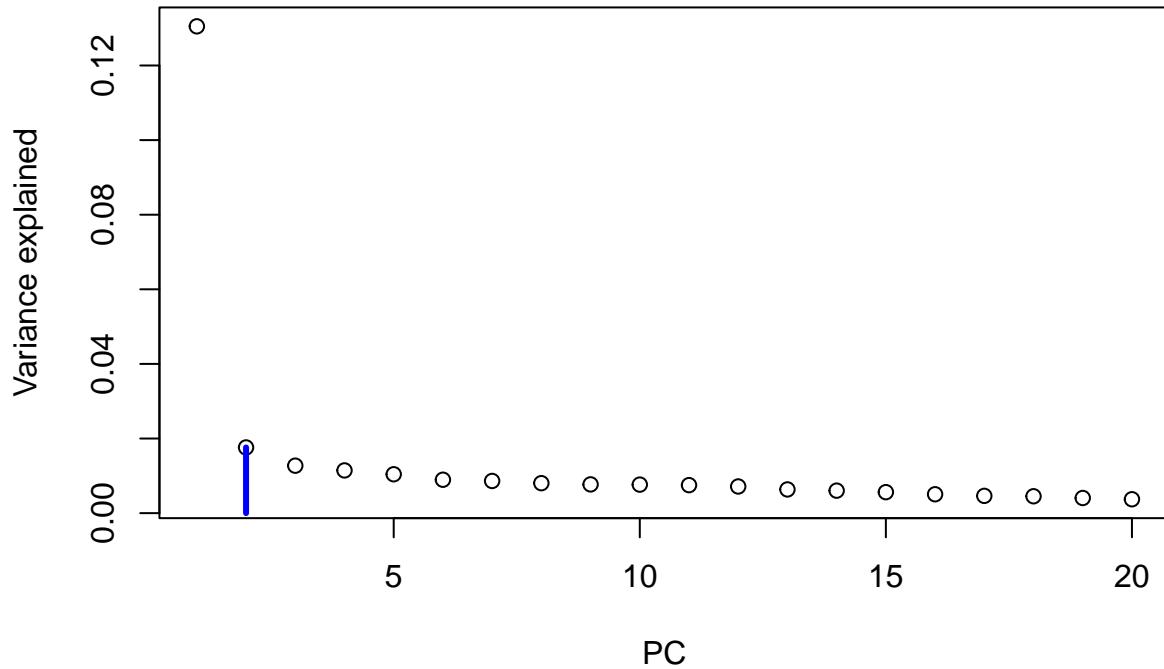
```

Perform PCA and find the number of latent factors

```

Y_23 <- as.matrix(geno_23[,-1])
pc_23 <- prcomp(Y_23)
plot(pc_23$sdev[1:20]^2, xlab = 'PC', ylab = "Variance explained")
points(2,pc_23$sdev[2]^2, type = "h", lwd = 3, col = "blue")

```



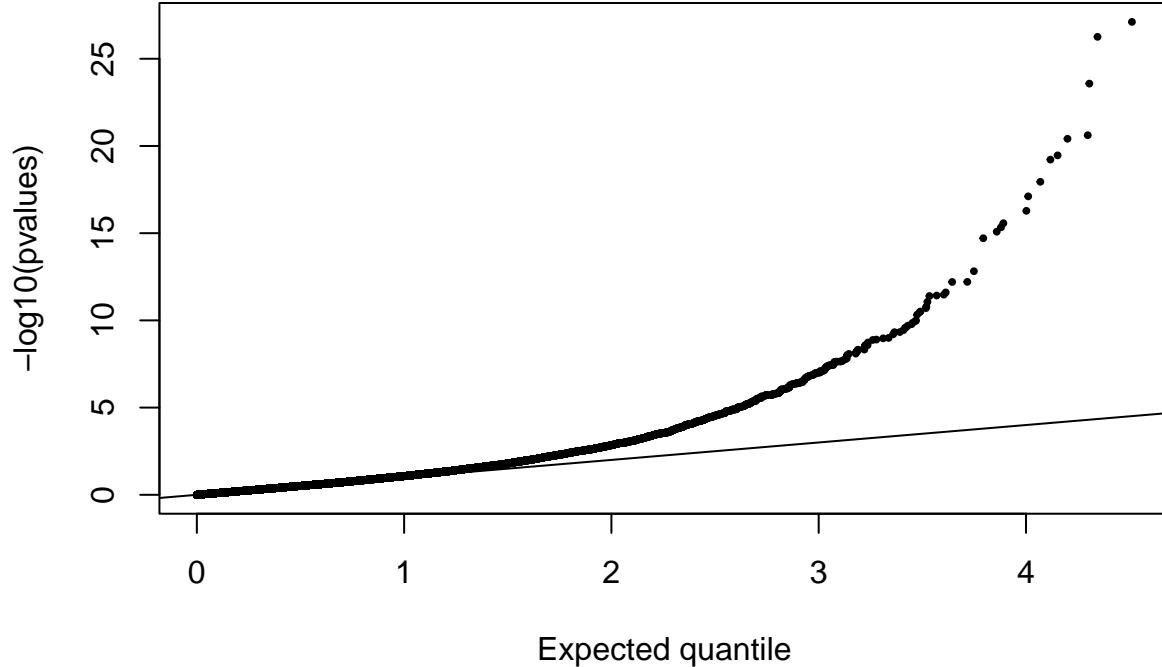
Fit the lfmm model and estimate the p value

```
#process the data
Y_23 <- as.matrix(geno_23[,-1])
Y_23 <- t(Y_23)
X_23 <- matrix(scale(pheno_23$blup_light), ncol = 1)
rownames(X_23) <- pheno_23$parent_pop
colnames(X_23) <- "blup_light"
common <- intersect(rownames(X_23), rownames(Y_23))
Y_23 <- Y_23[common,]
X_23 <- X_23[common, , drop = FALSE]

#fit the model
mod.lfmm_23 <- lfmm_ridge(Y = Y_23,
                           X = X_23,
                           K = 2)
pv_23 <- lfmm_test(Y = Y_23,
                     X = X_23,
                     lfmm = mod.lfmm_23,
                     calibrate = "gif")
pvalues_23 <- pv_23$calibrated.pvalue

#QQ plot
qqplot(rexp(length(pvalues), rate = log(10)),
       -log10(pvalues), xlab = "Expected quantile",
       pch = 19, cex = .4)
```

```
abline(0,1)
```

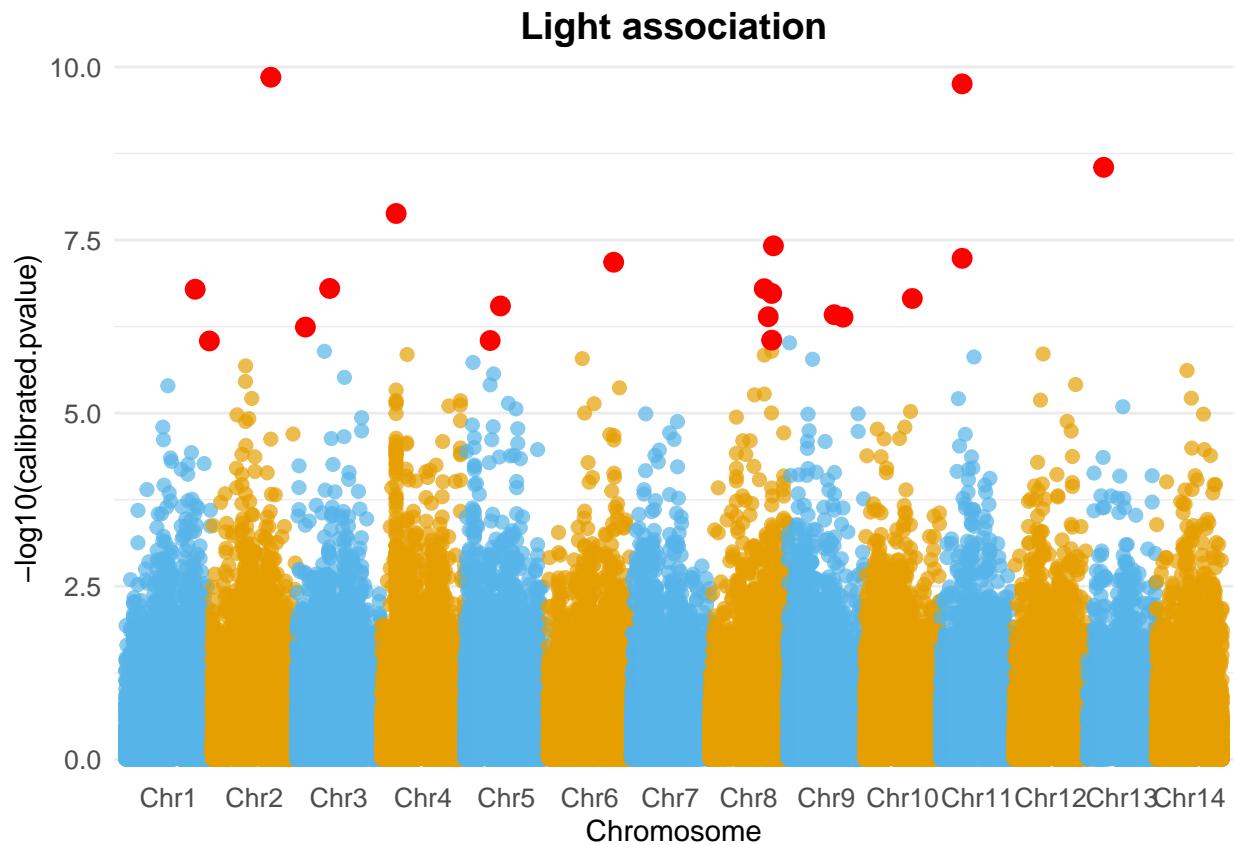


Create a data frame for plotting the manhattan plot.

```
plot_data_23 = tibble(loci = geno_23[,1], p_val = pv_23$pvalue[,1], calibrated.pvalue = pv_23$calibrated.pvalue) |> separate_wider_delim(loci, delim = ":" , names = c("chr", "bp")) |> filter(str_detect(chr, "Chr")) |> mutate(p.bonf = p.adjust(p_val, method = "bonferroni"), bp = as.numeric(bp))

manhattan_23 = plot_manhattan(plot_data_23,
                               chr_col = "chr",
                               pos_col = "bp",
                               value_col = "calibrated.pvalue",
                               transform = "neglog10",
                               title = "Light association",
                               highlight_top_n = 20)

print(manhattan_23)
```



```
# Save plot
ggsave("light_interaction23.png", manhattan_23, width = 12, height = 6, dpi = 300)
```

Manhattan plot side by side

```
library(cowplot)

##
## Attaching package: 'cowplot'

## The following object is masked from 'package:ggthemes':
##
##     theme_map

## The following object is masked from 'package:lubridate':
##
##     stamp

manhattan_tot <- plot_grid(manhattan_22, manhattan_23, ncol = 2, labels = c("2022", "2023"), rel_widths
```

list out top 20 SNPs for both the gardens

```

top20_UCD = plot_data |>
slice_min(calibrated.pvalue, n = 20) |>
select(seqid = chr, bp, p_val, calibrated.pvalue, p.bonf)
write_tsv(top20_UCD, "top20_UCD_loc.tsv")

top20_WL2 = plot_data_23 |>
slice_min(calibrated.pvalue, n = 20) |>
select(seqid = chr, bp, p_val, calibrated.pvalue, p.bonf)
write_tsv(top20_WL2, "top20_WL2_loc.tsv")

```

Plot the correlation of allele frequency and the blups

```

geno_table_23 = Y_23
colnames(geno_table_23) = geno_23[, 1]

freq_plot_data_23 = as.data.frame(X_23) |>
rownames_to_column(var = "Population") |>
mutate(loci1 = as.numeric(geno_table_23[, "Chr2:18026903"]),
loci2 = as.numeric(geno_table_23[, "Chr11:6546598"]),
loci3 = as.numeric(geno_table_23[, "Chr13:4965604"])) |>
pivot_longer(cols = loci1:loci3, names_to = "loci", values_to = "frequency")

freq_plot_data_23 |>
ggplot(aes(x = blup_light, y = frequency, label = Population)) +
geom_point(size = 3) +
geom_smooth(method = "lm", colour = "red") +
geom_text_repel(size = 6) +
facet_wrap(~loci, nrow = 1, scales = "free_y") +
theme_minimal()+
labs(x = "BLUP light", y = "Allele frequency")

## `geom_smooth()` using formula = 'y ~ x'

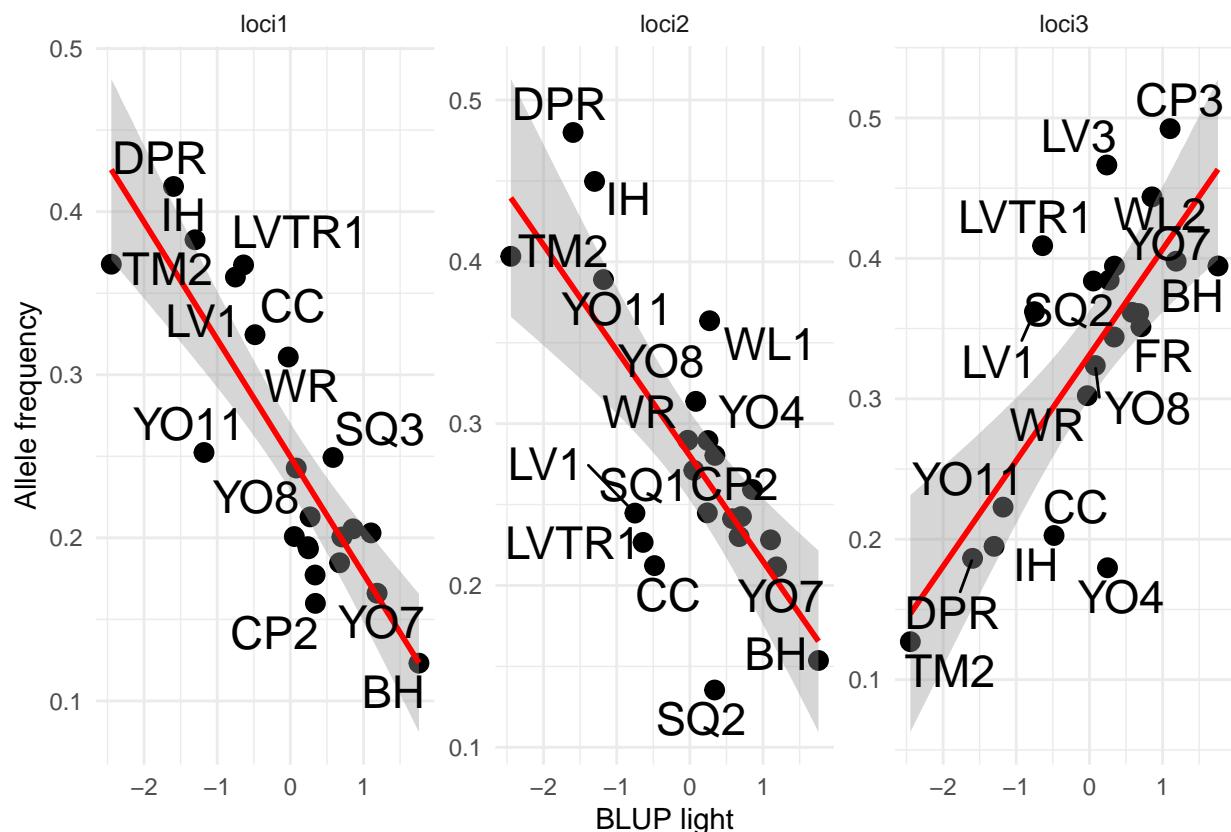
## Warning: The following aesthetics were dropped during statistical transformation: label.
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a 'group' aesthetic or to convert a numerical
##   variable into a factor?
## The following aesthetics were dropped during statistical transformation: label.
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a 'group' aesthetic or to convert a numerical
##   variable into a factor?
## The following aesthetics were dropped during statistical transformation: label.
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a 'group' aesthetic or to convert a numerical
##   variable into a factor?

## Warning: ggrepel: 9 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps

```

```
## Warning: ggrepel: 6 unlabeled data points (too many overlaps). Consider  
## increasing max.overlaps
```

```
## Warning: ggrepel: 5 unlabeled data points (too many overlaps). Consider  
## increasing max.overlaps
```



upload the gene table and find the nearby genes

```
genes = read_tsv("Data/gene_description.tsv")
```

```
## Rows: 34643 Columns: 6  
## -- Column specification -----  
## Delimiter: "\t"  
## chr (4): seqid, strand, gene_id, description  
## dbl (2): start, end  
##  
## i Use 'spec()' to retrieve the full column specification for this data.  
## i Specify the column types or set 'show_col_types = FALSE' to quiet this message.
```

```
window_size = 100000
```

```
genes_near_snps_UCD = top20_UCD |>  
  left_join(genes, by = "seqid", relationship = "many-to-many") |>  
  mutate(
```

```

# Calculate distances
dist_to_start = bp - start,
dist_to_end = bp - end,

# Check if SNP is within gene body
in_gene = bp >= start & bp <= end,

# Check if SNP is within window (upstream or downstream)
in_window = (bp >= (start - window_size) & bp <= (end + window_size))
) |>
filter(in_window) |>
mutate(
  # Categorize position
  position = case_when(
    in_gene ~ "in_gene",
    bp < start ~ "upstream",
    bp > end ~ "downstream"
  ),
  distance = case_when(
    in_gene ~ 0,
    bp < start ~ start - bp,
    bp > end ~ bp - end
  )
) |>
arrange(seqid, bp, distance)

write_tsv(genes_near_snps_UCD, "UCD_2022_genes.tsv")
genes_near_snps_UCD

```

```

## # A tibble: 511 x 16
##   seqid      bp     p_val calibrated.pvalue p.bonf  start      end strand gene_id
##   <chr>    <dbl>     <dbl>             <dbl>    <dbl>    <dbl>    <dbl> <chr>  <chr>
## 1 Chr1  102763 0.00000136        2.41e-21  0.131   99758  103380  +    Sdiv_p~
## 2 Chr1  102763 0.00000136        2.41e-21  0.131  103721  107316  -    Sdiv_p~
## 3 Chr1  102763 0.00000136        2.41e-21  0.131   98238  100014  -    Sdiv_p~
## 4 Chr1  102763 0.00000136        2.41e-21  0.131   92330   97787  -    Sdiv_p~
## 5 Chr1  102763 0.00000136        2.41e-21  0.131  108145  110992  +    Sdiv_p~
## 6 Chr1  102763 0.00000136        2.41e-21  0.131  109567  110014  -    Sdiv_p~
## 7 Chr1  102763 0.00000136        2.41e-21  0.131  111553  114116  +    Sdiv_p~
## 8 Chr1  102763 0.00000136        2.41e-21  0.131   85739   92723  +    Sdiv_p~
## 9 Chr1  102763 0.00000136        2.41e-21  0.131  115206  121670  -    Sdiv_p~
## 10 Chr1 102763 0.00000136        2.41e-21  0.131  118937  119409  +    Sdiv_p~
## # i 501 more rows
## # i 7 more variables: description <chr>, dist_to_start <dbl>,
## #   dist_to_end <dbl>, in_gene <lgl>, in_window <lgl>, position <chr>,
## #   distance <dbl>

genes_near_snps_WL2 = top20_WL2 |>
  left_join(genes, by = "seqid", relationship = "many-to-many") |>
  mutate(
    # Calculate distances
    dist_to_start = bp - start,
    dist_to_end = bp - end,

```

```

# Check if SNP is within gene body
in_gene = bp >= start & bp <= end,

# Check if SNP is within window (upstream or downstream)
in_window = (bp >= (start - window_size) & bp <= (end + window_size))
) |>
filter(in_window) |>
mutate(
  # Categorize position
  position = case_when(
    in_gene ~ "in_gene",
    bp < start ~ "upstream",
    bp > end ~ "downstream"
  ),
  distance = case_when(
    in_gene ~ 0,
    bp < start ~ start - bp,
    bp > end ~ bp - end
  )
) |>
arrange(seqid, bp, distance)

write_tsv(genes_near_snps_WL2, "WL2_2023_genes.tsv")
genes_near_snps_WL2

```

```

## # A tibble: 278 x 16
##   seqid      bp   p_val calibrated.pvalue p.bonf  start     end strand gene_id
##   <chr>    <dbl>     <dbl>           <dbl>   <dbl>   <dbl> <dbl> <chr>  <chr>
## 1 Chr1  21374778 1.39e-5     0.000000163     1 2.14e7 2.14e7  +    Sdiv_p~
## 2 Chr1  21374778 1.39e-5     0.000000163     1 2.14e7 2.14e7  +    Sdiv_p~
## 3 Chr1  21374778 1.39e-5     0.000000163     1 2.14e7 2.14e7  +    Sdiv_p~
## 4 Chr1  21374778 1.39e-5     0.000000163     1 2.14e7 2.14e7  -    Sdiv_p~
## 5 Chr1  21374778 1.39e-5     0.000000163     1 2.14e7 2.14e7  -    Sdiv_p~
## 6 Chr1  21374778 1.39e-5     0.000000163     1 2.14e7 2.14e7  -    Sdiv_p~
## 7 Chr1  21374778 1.39e-5     0.000000163     1 2.14e7 2.14e7  +    Sdiv_p~
## 8 Chr1  21374778 1.39e-5     0.000000163     1 2.14e7 2.14e7  -    Sdiv_p~
## 9 Chr1  21374778 1.39e-5     0.000000163     1 2.14e7 2.14e7  +    Sdiv_p~
## 10 Chr1 21374778 1.39e-5     0.000000163     1 2.13e7 2.14e7  +    Sdiv_p~
## # i 268 more rows
## # i 7 more variables: description <chr>, dist_to_start <dbl>,
## #   dist_to_end <dbl>, in_gene <lgl>, in_window <lgl>, position <chr>,
## #   distance <dbl>

```

```

prioritized_genes <- genes_near_snps_UCD %>%
  group_by(gene_id, description, seqid) %>%
  summarise(
    n_snps = n(),
    min_distance = min(distance),
    any_in_gene = any(position == "in_gene"),
    any_promoter = any(distance <= 1000 & position=="upstream"),
    .groups = "drop"
  ) %>%

```

```

mutate(
  priority_score =
    3 * any_in_gene +
    2 * any_promoter +
    1 * (min_distance <= 3000) +
    n_snps
) %>%
arrange(desc(priority_score), min_distance)

prioritized_genes

## # A tibble: 358 x 8
##   gene_id      description seqid n_snps min_distance any_in_gene any_promoter
##   <chr>        <chr>      <chr>  <int>       <dbl> <lgl>      <lgl>
## 1 Sdiv_ptg00001~ Similar to~ Chr4     5          0 TRUE      FALSE
## 2 Sdiv_ptg00001~ Similar to~ Chr1     2          0 TRUE      FALSE
## 3 Sdiv_ptg00001~ Similar to~ Chr1     2          0 TRUE      FALSE
## 4 Sdiv_ptg00001~ Similar to~ Chr4     5         1652 FALSE     FALSE
## 5 Sdiv_ptg00000~ Similar to~ Chr11    1          0 TRUE      FALSE
## 6 Sdiv_ptg00000~ Protein of~ Chr3     1          0 TRUE      FALSE
## 7 Sdiv_ptg00000~ Similar to~ Chr9     1          0 TRUE      FALSE
## 8 Sdiv_ptg00000~ Similar to~ Chr14    1          0 TRUE      FALSE
## 9 Sdiv_ptg00000~ Similar to~ Chr2     1          0 TRUE      FALSE
## 10 Sdiv_ptg00001~ Protein of~ Chr1    2          698 FALSE     TRUE
## # i 348 more rows
## # i 1 more variable: priority_score <dbl>

plot_locus_zoom <- function(snp_chr, snp_bp, snp_table, gene_table, window = 50000) {

  region_start <- snp_bp - window
  region_end   <- snp_bp + window

  # subset SNPs
  snps_region <- snp_table %>%
    filter(chr == snp_chr, bp >= region_start, bp <= region_end)

  # subset genes
  genes_region <- gene_table %>%
    filter(seqid == snp_chr, start <= region_end, end >= region_start)

  # panel 1: SNP p-values
  p1 <- ggplot(snps_region, aes(x = bp, y = -log10(calibrated.pvalue))) +
    geom_point(color = "black") +
    geom_point(data = snps_region %>% filter(bp == snp_bp),
               aes(x = bp, y = -log10(calibrated.pvalue)),
               color = "red", size = 3) +
    theme_bw() +
    labs(title = paste0("LocusZoom: ", snp_chr, ":", snp_bp),
         y = "-log10(p)", x = "")

  # panel 2: gene structure
  p2 <- ggplot() +
    geom_segment(

```

```

        data = genes_region,
        aes(x = start, xend = end, y = 0, yend = 0),
        size = 5, color = "skyblue"
    ) +
    geom_point(
        aes(x = snp_bp, y = 0),
        color = "red", size = 4
    ) +
    labs(y = "Genes", x = "Position (bp)") +
    theme_bw() +
    theme(
        axis.text.y = element_blank(),
        axis.ticks.y = element_blank()
    )

    list(p1 = p1, p2 = p2)
}

zoom <- plot_locus_zoom(
    snp_chr = "Chr1",
    snp_bp = 102763,
    snp_table = plot_data,
    gene_table = genes,
    window = 50000
)

```

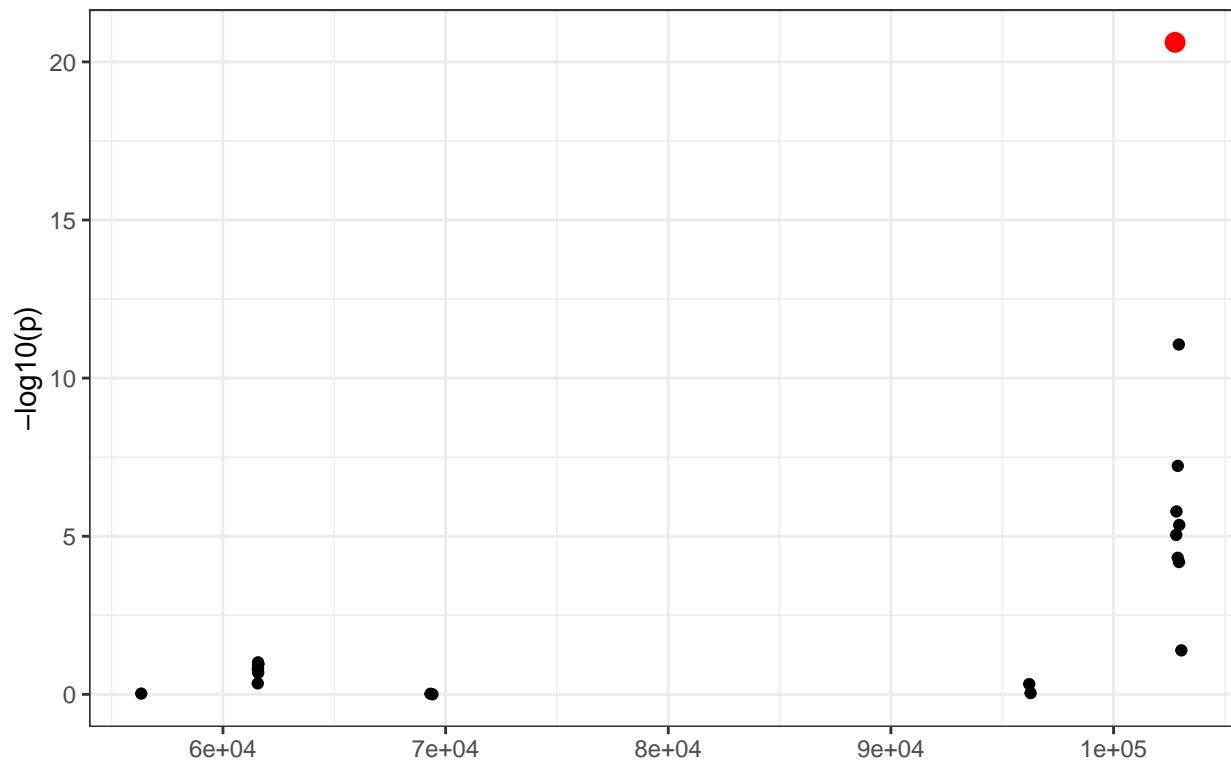
Warning: Using ‘size’ aesthetic for lines was deprecated in ggplot2 3.4.0.
i Please use ‘linewidth’ instead.
This warning is displayed once every 8 hours.
Call ‘lifecycle::last_lifecycle_warnings()’ to see where this warning was
generated.

```

zoom$p1    # SNP plot

```

LocusZoom: Chr1:102763



```
zoom$p2    # Gene track
```

