main

July 13, 2021

1 Differential Expression with limma-voom pipeline - Genes

[1]:
```
suppressMessages({library(SummarizedExperiment)
                  library(data.table)
                  library(tidyverse)
                  library(memoise)
                  library(limma)
                  library(edgeR)
                  library(sva)})
```

1.1 Prepare Data

1.2 Functions

1.2.1 General functions

[2]:
```
# Function from jaffelab github
merge_rse_metrics <- function(rse) {
    stopifnot(is(rse, 'RangedSummarizedExperiment'))
    stopifnot(
        c('concordMapRate', 'overallMapRate', 'mitoRate', 'rRNA_rate',
          'totalAssignedGene', 'numMapped', 'numReads', 'numUnmapped',
          'mitoMapped', 'totalMapped') %in%
            colnames(SummarizedExperiment::colData(rse))
    )

    stopifnot(all(sapply(c(
        'concordMapRate', 'overallMapRate', 'mitoRate', 'rRNA_rate',
        'totalAssignedGene', 'numMapped', 'numReads', 'numUnmapped',
        'mitoMapped', 'totalMapped'), function(var) {
            is(colData(rse)[, var], 'List')
        })
    ))

    rse$concordMapRate = mapply(function(r, n) {
        sum(r*n)/sum(n)
    }, rse$concordMapRate, rse$numReads)
    rse$overallMapRate = mapply(function(r, n) {
```

```
        sum(r*n)/sum(n)
    }, rse$overallMapRate, rse$numReads)
    rse$mitoRate = mapply(function(r, n) {
        sum(r*n)/sum(n)
    }, rse$mitoRate, rse$numMapped)
    rse$rRNA_rate = mapply(function(r, n) {
        sum(r*n)/sum(n)
    }, rse$rRNA_rate, rse$numMapped)
    rse$totalAssignedGene = mapply(function(r, n) {
        sum(r*n)/sum(n)
    }, rse$totalAssignedGene, rse$numMapped)

    rse$numMapped = sapply(rse$numMapped, sum)
    rse$numReads = sapply(rse$numReads, sum)
    rse$numUnmapped = sapply(rse$numUnmapped, sum)
    rse$mitoMapped = sapply(rse$mitoMapped, sum)
    rse$totalMapped = sapply(rse$totalMapped, sum)
    return(rse)
}
```

[3]:
```
save_volcanoPlot <- function(top, label, dirname){
    pdf(file=paste0(dirname, "/volcanoPlot_", label, ".pdf"), 8, 6)
    with(top, plot(logFC, -log10(P.Value), pch=20, cex=0.6))
    with(subset(top, adj.P.Val<=0.05), points(logFC, -log10(P.Value),
                                              pch=20, col='red', cex=0.6))
    with(subset(top, abs(logFC)>0.50), points(logFC, -log10(P.Value),
                                              pch=20, col='orange', cex=0.6))
    with(subset(top, adj.P.Val<=0.05 & abs(logFC)>0.50),
         points(logFC, -log10(P.Value), pch=20, col='green', cex=0.6))
    dev.off()
}

save_MAplot <- function(top, label, dirname){
    pdf(file=paste0(dirname, "/MAplot_", label, ".pdf"), 8, 6)
    with(top, plot(AveExpr, logFC, pch=20, cex=0.5))
    with(subset(top, adj.P.Val<0.05),
         points(AveExpr, logFC, col="red", pch=20, cex=0.5))
    dev.off()
}

extract_de <- function(contrast, label, efit, seed_int){
    dirname = paste0("permutation_", stringr::str_pad(seed_int, 2, pad = "0"))
    top <- topTable(efit, coef=contrast, number=Inf, sort.by="P")
    top <- top[order(top$P.Value), ]
    top.fdr <- top %>% filter(adj.P.Val<=0.05)
    print(paste("Comparison for:", label))
    print(paste('There are:', dim(top.fdr)[1], 'DE features!'))
```

```
    fwrite(top,
           file=paste0(dirname, "/diffExpr_", label, "_full.txt"),
           sep='\t', row.names=TRUE)
    fwrite(top.fdr,
           file=paste0(dirname, "/diffExpr_", label, "_FDR05.txt"),
           sep='\t', row.names=TRUE)
    save_volcanoPlot(top, label, dirname)
    save_MAplot(top, label, dirname)
}
```

1.2.2 Cached functions

[4]:
```
load_counts <- function(){
    counts_file = paste0("/ceph/projects/v3_phase3_paper/inputs/phase2/_m/
 ↪count_data/",
                         ␣
 ↪"dlpfc_ribozero_brainseq_phase2_hg38_rseGene_merged_n453.rda")
    load(counts_file)
    rse_df = rse_gene
    return(rse_df)
}
memCounts <- memoise(load_counts)

get_random_samples <- function(seed_int, new_dir=TRUE){
    set.seed(seed_int + 113) # seed for reproducibility
    dirname = paste0("permutation_", stringr::str_pad(seed_int, 2, pad = "0"))
    if(new_dir){
        dir.create(dirname)
    }
    rse_df <- memCounts()
    keepIndex = which((rse_df$Dx %in% c("Control", "Schizo")) &
                      (rse_df$Age > 17) & (rse_df$Sex == "M") &
                      (rse_df$Race %in% c("AA", "CAUC")))
    snames = sample(keepIndex, 114, replace=FALSE) # subsampling to Female N␣
 ↪(sample size)
    return(snames)
}
memSamples <- memoise(get_random_samples)

get_MDS_genotypes <- function(){
    mds_file = paste0("/ceph/projects/v3_phase3_paper/inputs/genotypes/to_brnum/
 ↪",
                      "merge/to_plink/mds/_m/merged.mds")
    mds = fread(mds_file) %>%
        rename("snpPC1"="C1", "snpPC2"="C2", "snpPC3"="C3",
               "snpPC4"="C4", "snpPC5"="C5") %>%
        mutate_if(is.character, as.factor)
```

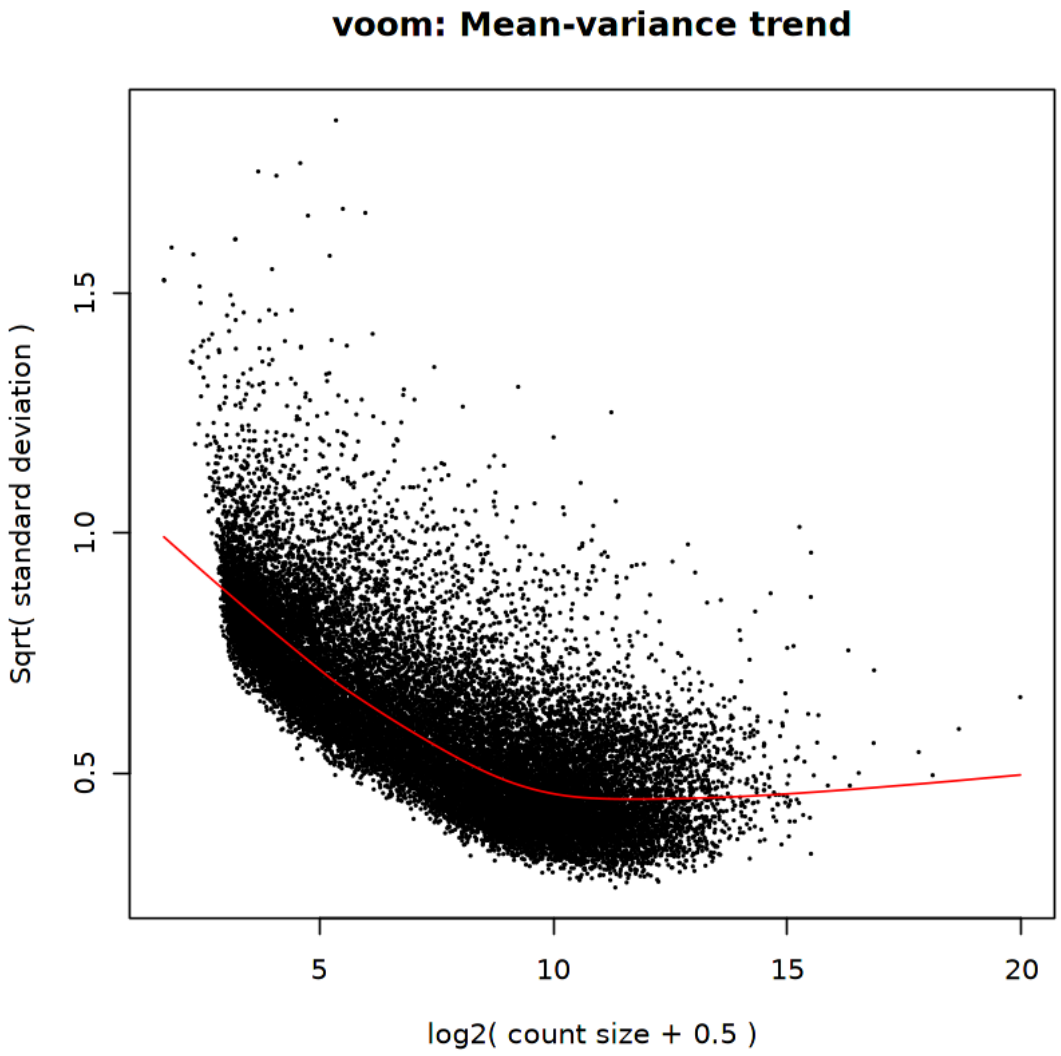
```
    return(mds)
}
memMDS <- memoise(get_MDS_genotypes)

prep_data <- function(seed_int){
    rse_df <- memCounts()
    keepIndex <- memSamples(seed_int)
    rse_df = rse_df[, keepIndex]
    rse_df$Dx = factor(rse_df$Dx, levels = c("Control", "Schizo"))
    rse_df$Sex <- factor(rse_df$Sex)
    rse_df <- merge_rse_metrics(rse_df)
    rse_df$ERCCsumLogErr <- mapply(function(r, n) {
        sum(r * n)/sum(n)
    }, rse_df$ERCCsumLogErr, rse_df$numReads)
    colData(rse_df)$RIN = sapply(colData(rse_df)$RIN,"[",1)
    rownames(colData(rse_df)) <- sapply(strsplit(rownames(colData(rse_df)),␣
 ↪"_"), "[", 1)
    pheno = colData(rse_df) %>% as.data.frame %>%
        inner_join(memMDS(), by=c("BrNum"="FID"))
    # Generate DGE list
    x <- DGEList(counts=assays(rse_df)$counts,
                 genes=rowData(rse_df),
                 samples=pheno)
    # Filter by expression
    design0 <- model.matrix(~Dx, data=x$samples)
    keep.x <- filterByExpr(x, design=design0)
    x <- x[keep.x, , keep.lib.sizes=FALSE]
    print(paste('There are:', sum(keep.x), 'features left!', sep=' '))
    # Normalize library size
    x <- calcNormFactors(x, method="TMM")
    return(x)
}
memo_prepData <- memoise(prep_data)

SVA_model <- function(seed_int){
    x <- memo_prepData(seed_int)
    # Design matrix
    mod = model.matrix(~Dx + Age + mitoRate + rRNA_rate + RIN +
                       totalAssignedGene + overallMapRate + ERCCsumLogErr +
                       snpPC1 + snpPC2 + snpPC3, data=x$samples)
    colnames(mod) <- gsub("Dx", "", colnames(mod))
    colnames(mod) <- gsub("\\(Intercept\\)", "Intercept", colnames(mod))
    # Calculated SVs
    null.model = mod %>% as.data.frame %>% select(-c("Schizo")) %>% as.matrix
    n.sv <- num.sv(x$counts, mod, method="be")
    svobj <- svaseq(x$counts, mod, null.model, n.sv=n.sv)
    if(svobj$sv == 0){
```

```
        modQsva <- mod
    } else {
        modQsva <- cbind(mod, svobj$sv)
        len.d <- length(colnames(modQsva))
        colnames(modQsva)[((len.d - n.sv)+1):len.d] <- make.names(paste0("sv",1:
 ↪n.sv))
    }
    return(modQsva)
}
memo_svaModel <- memoise(SVA_model)

get_voom <- function(seed_int){
    ### Preform voom
    x <- memo_prepData(seed_int)
    modQsva <- memo_svaModel(seed_int)
    v <- voom(x[, rownames(modQsva)], modQsva, plot=TRUE)
    return(v)
}
memo_voom <- memoise(get_voom)

cal_res <- function(seed_int){
    ### Calculate residuals
    v <- memo_voom(seed_int)
    null_model <- v$design %>% as.data.frame %>% select(-c("Schizo")) %>% as.
 ↪matrix
    fit_res <- lmFit(v, design=null_model)
    res = v$E - ( fit_res$coefficients %*% t(null_model) )
    res_sd = apply(res, 1, sd)
    res_mean = apply(res, 1, mean)
    res_norm = (res - res_mean) / res_sd
    dirname = paste0("permutation_", stringr::str_pad(seed_int, 2, pad = "0"))
    write.table(res_norm, file=paste0(dirname, '/residualized_expression.tsv'),
                sep="\t", quote=FALSE)
}
memo_res <- memoise(cal_res)

fit_voom <- function(seed_int){
    v <- memo_voom(seed_int)
    modQsva <- memo_svaModel(seed_int)
    fit0 <- lmFit(v, modQsva)
    contr.matrix <- makeContrasts(CtrlvsSZ = Schizo,
                                  levels=colnames(modQsva))
    fit <- contrasts.fit(fit0, contrasts=contr.matrix)
    esv <- eBayes(fit)
    return(esv)
}
memo_efit <- memoise(fit_voom)
```

1.3 Differential Expression Analysis

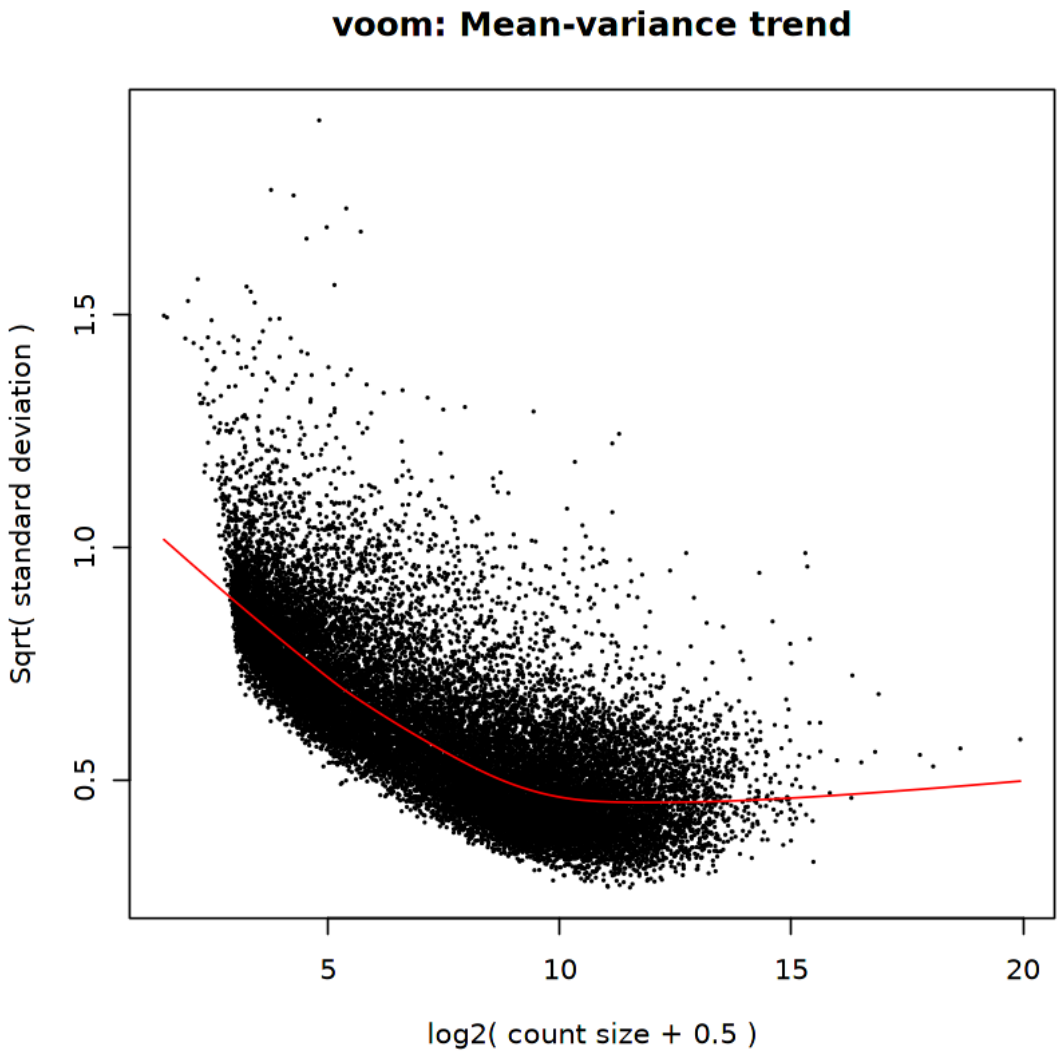
[5]:
```
for(seed_int in seq(1, 10)){
    # Preform voom
    v <- memo_voom(seed_int)
    dirname = paste0("permutation_", stringr::str_pad(seed_int, 2, pad = "0"))
    save(v, file=paste0(dirname,'/voomSVA.RData'))
    # Fit model and apply eBayes
    efit = memo_efit(seed_int)
    # Save differential expression
    extract_de(1, "CtrlvsSZ", efit, seed_int)
    # Calculate residuals
    memo_res(seed_int)
}
```

```
[1] "There are: 22545 features left!"
Number of significant surrogate variables is:  1
Iteration (out of 5 ):1  2  3  4  5

Warning message in if (svobj$sv == 0) {:
"the condition has length > 1 and only the first element will be used"

[1] "Comparison for: CtrlvsSZ"
[1] "There are: 1 DE features!"
[1] "There are: 22470 features left!"
Number of significant surrogate variables is:  1
Iteration (out of 5 ):1  2  3  4  5

Warning message in if (svobj$sv == 0) {:
"the condition has length > 1 and only the first element will be used"

[1] "Comparison for: CtrlvsSZ"
[1] "There are: 0 DE features!"
[1] "There are: 22378 features left!"
Number of significant surrogate variables is:  1
Iteration (out of 5 ):1  2  3  4  5

Warning message in if (svobj$sv == 0) {:
"the condition has length > 1 and only the first element will be used"
```



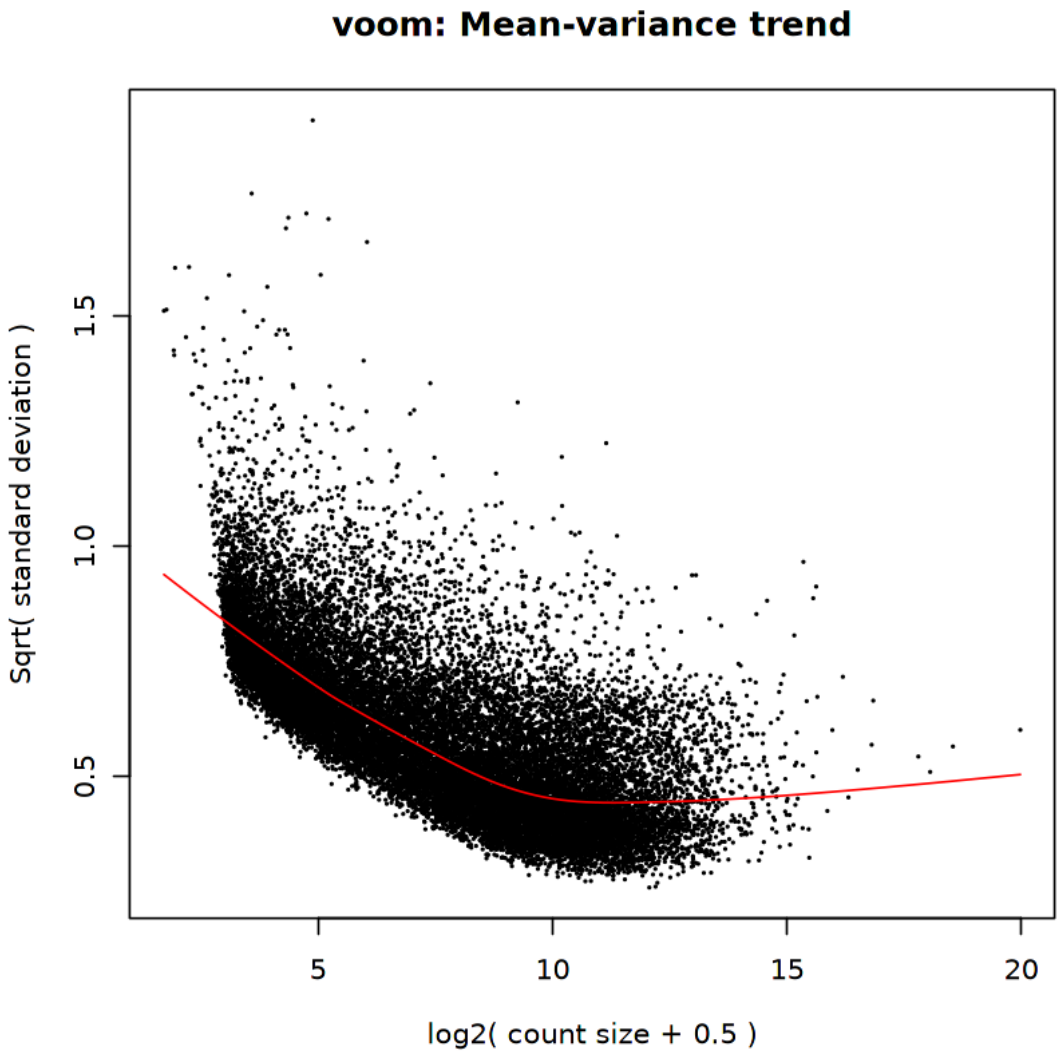

```
[1] "Comparison for: CtrlvsSZ"
[1] "There are: 0 DE features!"
[1] "There are: 22560 features left!"
Number of significant surrogate variables is:  1
Iteration (out of 5 ):1  2  3  4  5
```

```
Warning message in if (svobj$sv == 0) {:
"the condition has length > 1 and only the first element will be used"
```



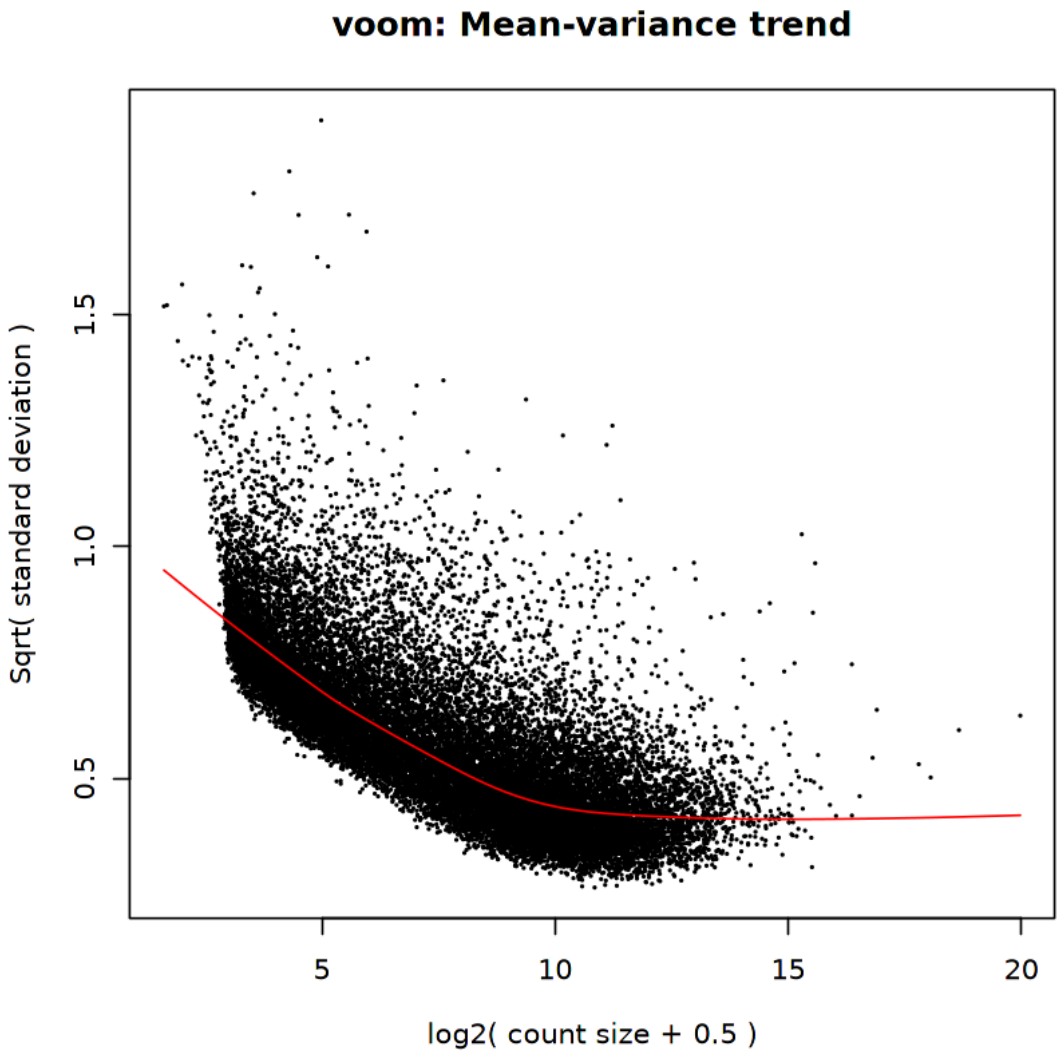

```
[1] "Comparison for: CtrlvsSZ"
[1] "There are: 309 DE features!"
[1] "There are: 22638 features left!"
Number of significant surrogate variables is:  1
Iteration (out of 5 ):1  2  3  4  5
```

```
Warning message in if (svobj$sv == 0) {:
"the condition has length > 1 and only the first element will be used"
```



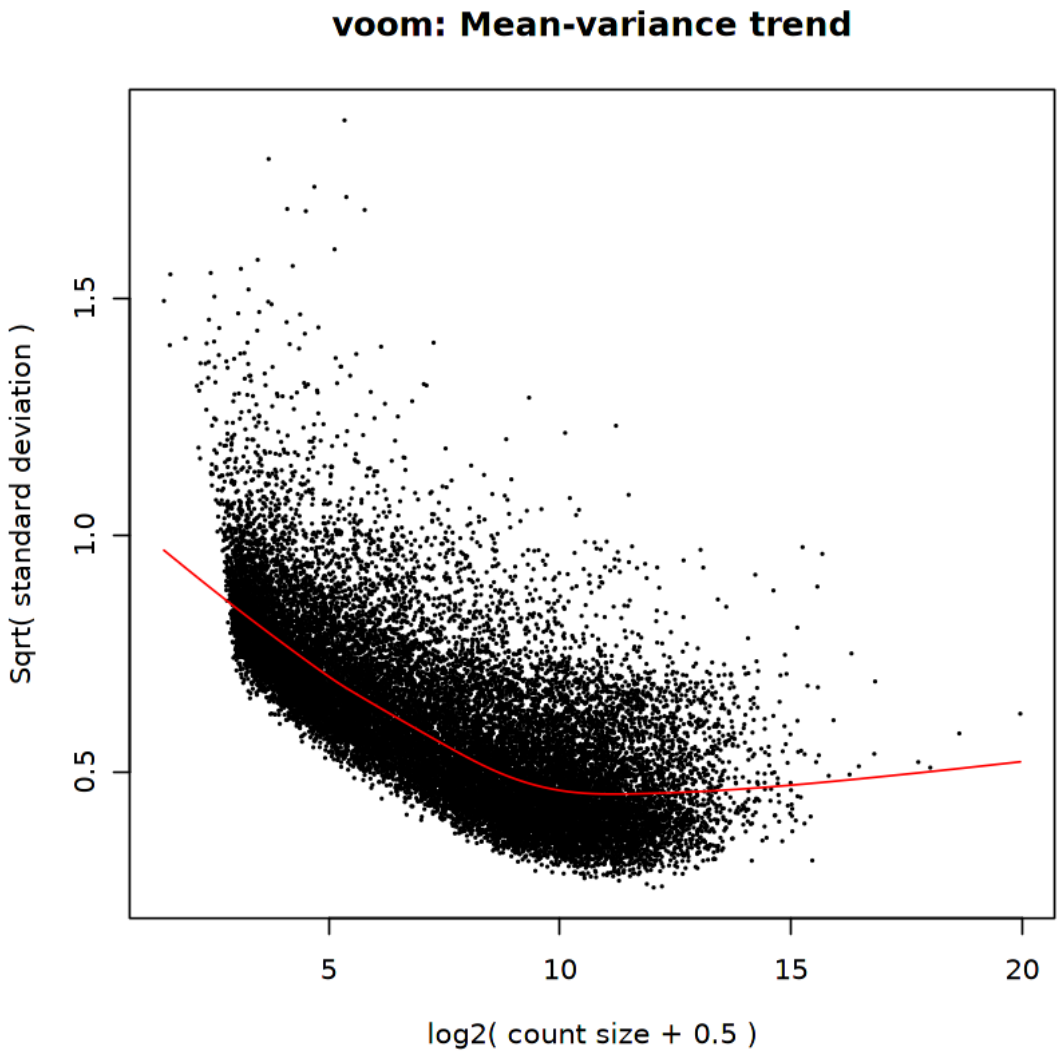

```
[1] "Comparison for: CtrlvsSZ"
[1] "There are: 6 DE features!"
[1] "There are: 22376 features left!"
Number of significant surrogate variables is:  1
Iteration (out of 5 ):1  2  3  4  5
```

```
Warning message in if (svobj$sv == 0) {:
"the condition has length > 1 and only the first element will be used"
```



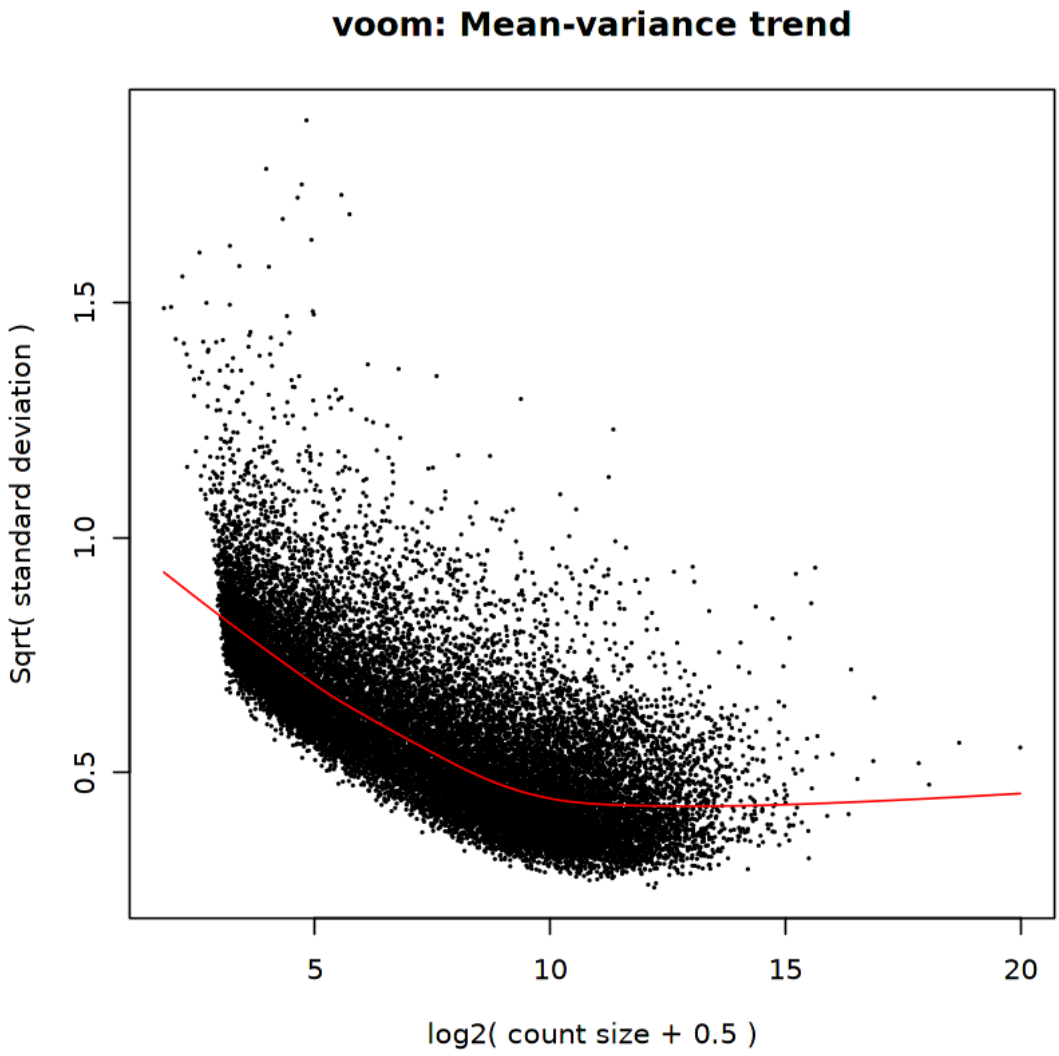

```
[1] "Comparison for: CtrlvsSZ"
[1] "There are: 4 DE features!"
[1] "There are: 22198 features left!"
Number of significant surrogate variables is:  1
Iteration (out of 5 ):1  2  3  4  5
```

```
Warning message in if (svobj$sv == 0) {:
"the condition has length > 1 and only the first element will be used"
```



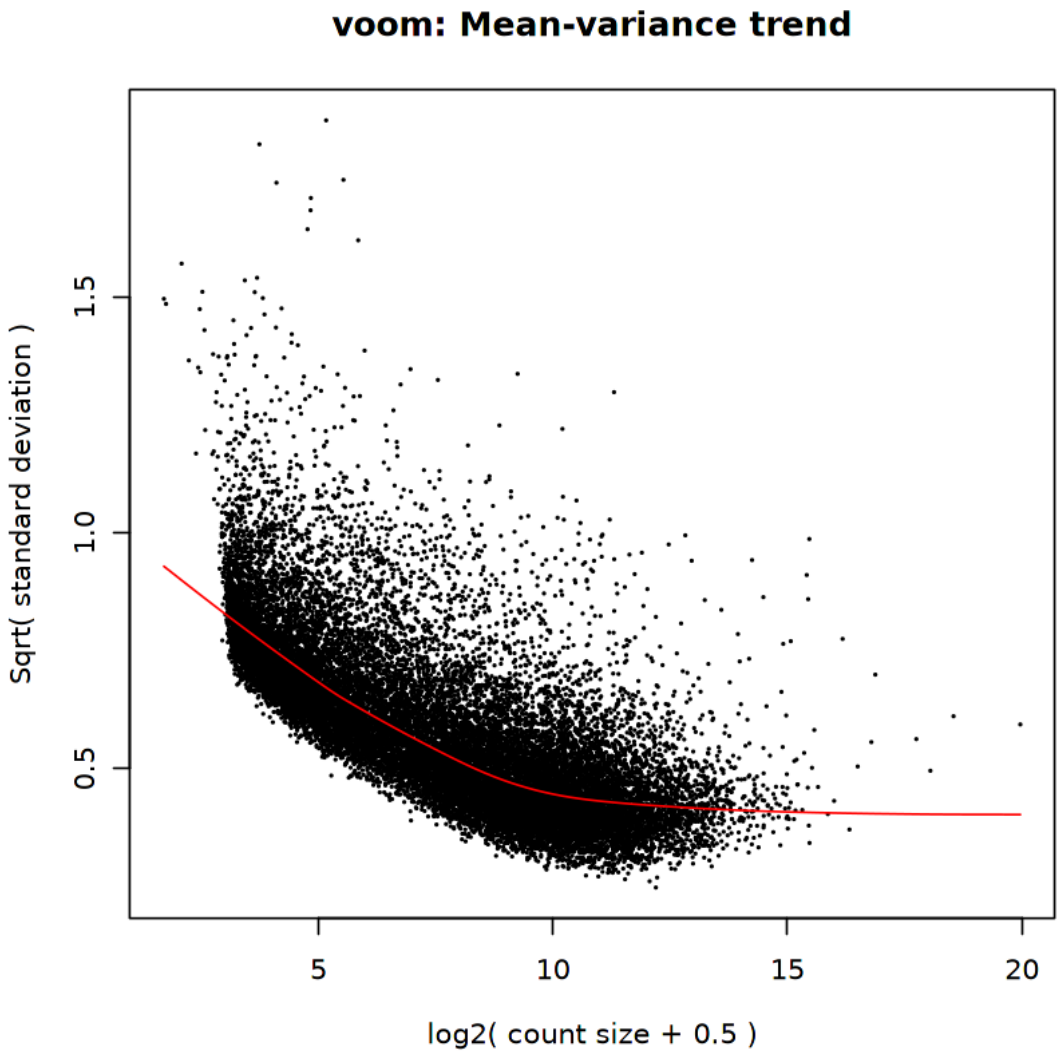

```
[1] "Comparison for: CtrlvsSZ"
[1] "There are: 3 DE features!"
[1] "There are: 22349 features left!"
Number of significant surrogate variables is:  1
Iteration (out of 5 ):1  2  3  4  5
```

```
Warning message in if (svobj$sv == 0) {:
"the condition has length > 1 and only the first element will be used"
```



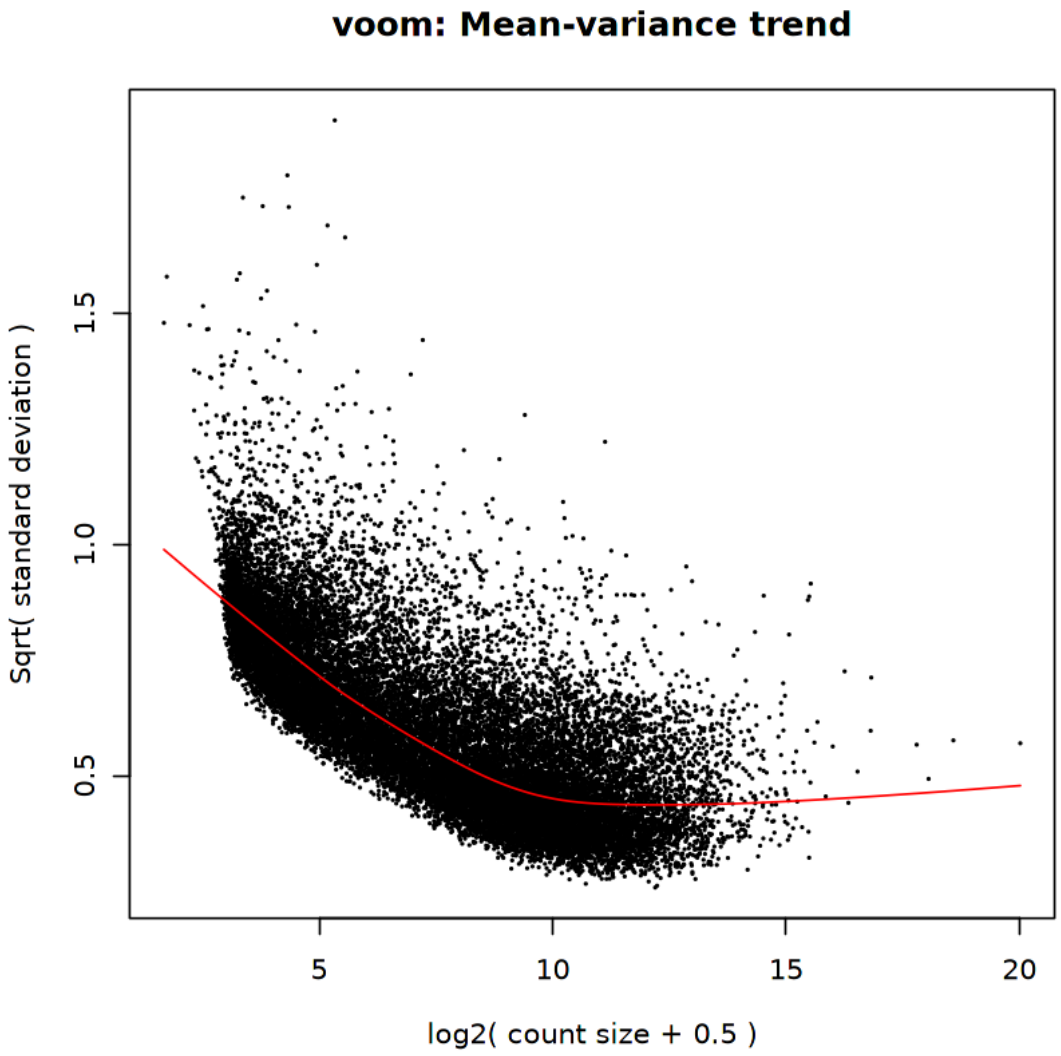

```
[1] "Comparison for: CtrlvsSZ"
[1] "There are: 152 DE features!"
[1] "There are: 22378 features left!"
Number of significant surrogate variables is:  1
Iteration (out of 5 ):1  2  3  4  5
```

```
Warning message in if (svobj$sv == 0) {:
"the condition has length > 1 and only the first element will be used"
```

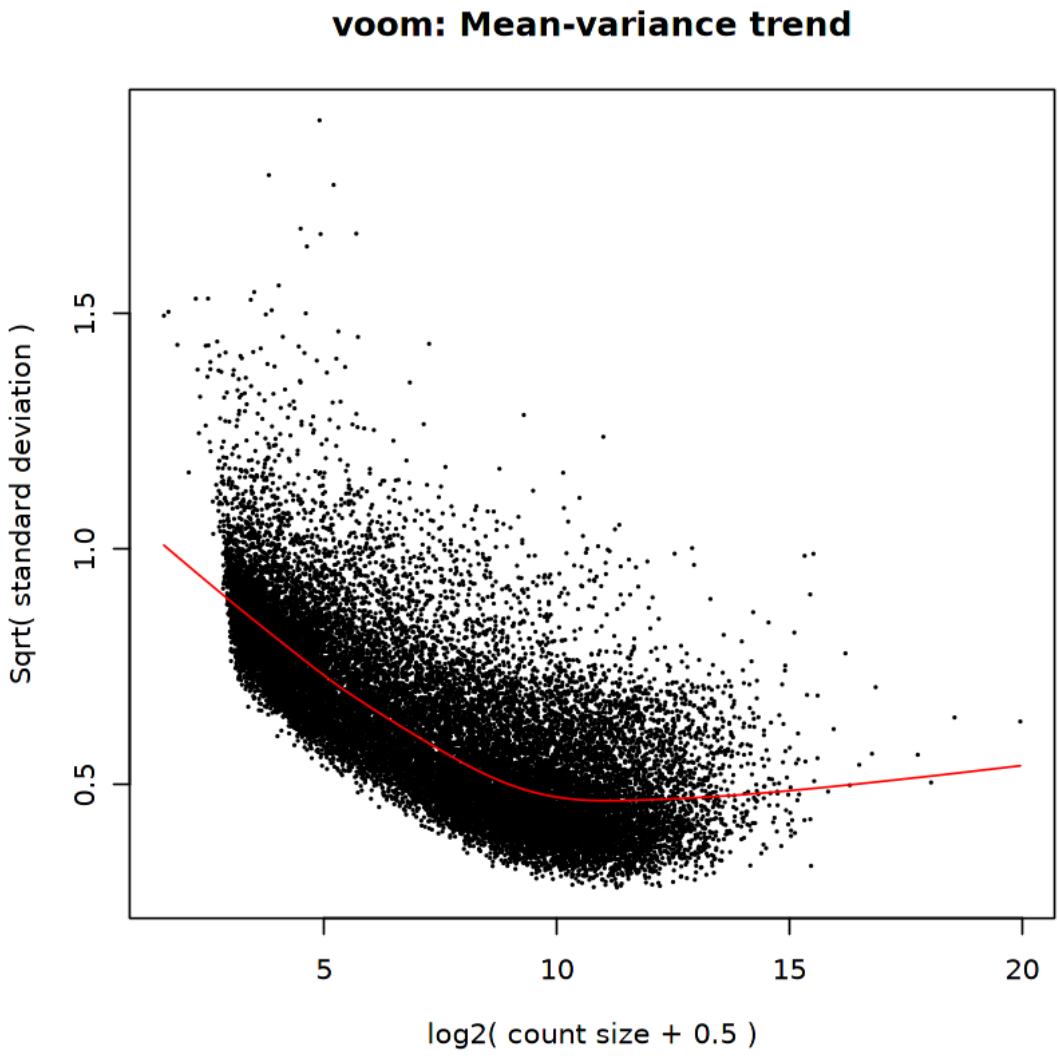


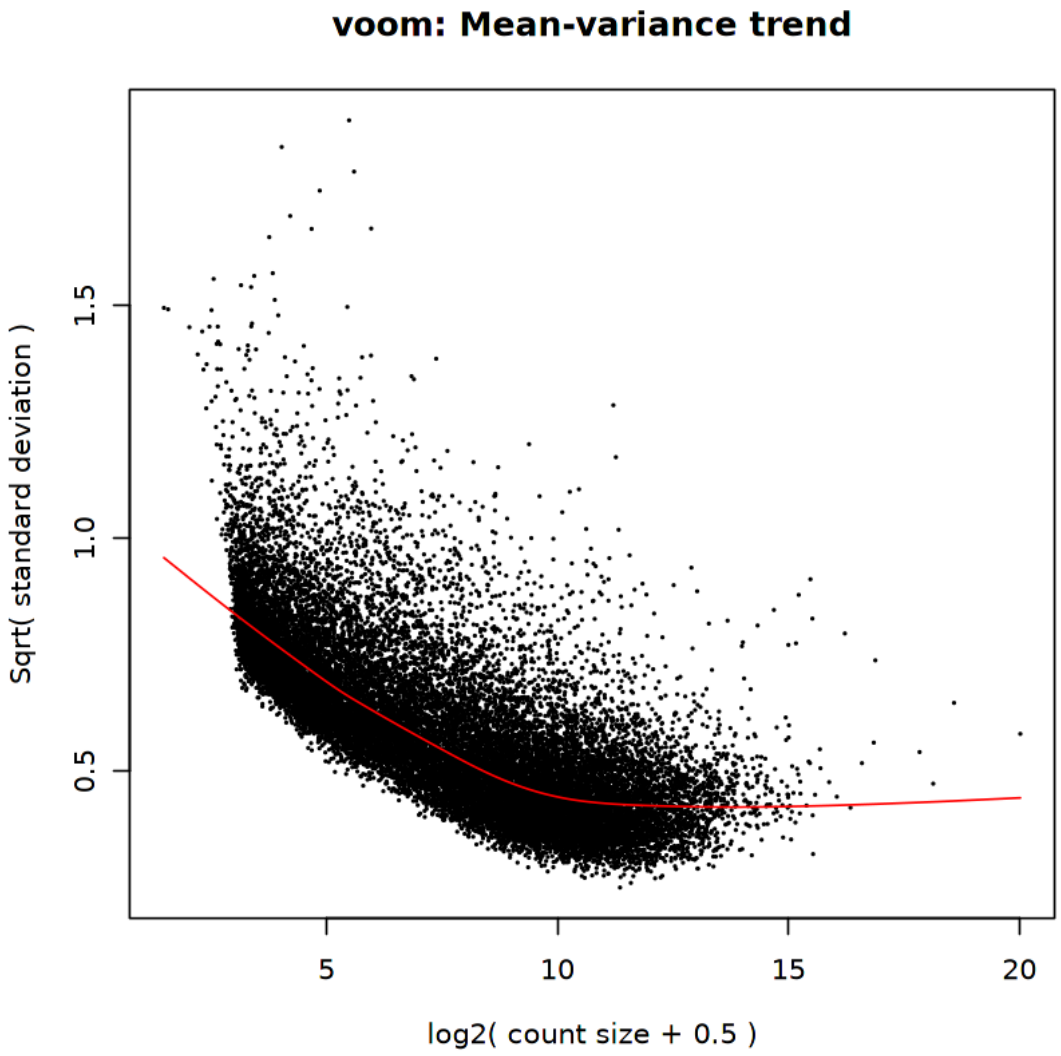

```
[1] "Comparison for: CtrlvsSZ"
[1] "There are: 2 DE features!"
[1] "There are: 22517 features left!"
Number of significant surrogate variables is:  1
Iteration (out of 5 ):1  2  3  4  5
```

```
Warning message in if (svobj$sv == 0) {:
"the condition has length > 1 and only the first element will be used"
```




```
[1] "Comparison for: CtrlvsSZ"
[1] "There are: 4 DE features!"
```


voom: Mean-variance trend
Sqrt(standard deviation)
0.5
1.0
1.5
5
10
15
20
log2(count size + 0.5)



1.4 Repreducibility Information

[6]:
```
Sys.time()
proc.time()
options(width = 120)
sessioninfo::session_info()
```

```
[1] "2021-07-13 15:59:33 EDT"
```

```
    user   system  elapsed
2293.522  549.637  527.325
```

```
 Session info
 setting  value
```

```
version  R version 4.0.3 (2020-10-10)
os       Arch Linux
system   x86_64, linux-gnu
ui       X11
language (EN)
collate  en_US.UTF-8
ctype    en_US.UTF-8
tz       America/New_York
date     2021-07-13

 Packages
package              * version  date       lib source
annotate               1.68.0   2020-10-27 [1] Bioconductor
AnnotationDbi          1.52.0   2020-10-27 [1] Bioconductor
assertthat             0.2.1    2019-03-21 [1] CRAN (R 4.0.2)
backports              1.2.1    2020-12-09 [1] CRAN (R 4.0.2)
base64enc              0.1-3    2015-07-28 [1] CRAN (R 4.0.2)
Biobase              * 2.50.0   2020-10-27 [1] Bioconductor
BiocGenerics         * 0.36.1   2021-04-16 [1] Bioconductor
BiocParallel         * 1.24.1   2020-11-06 [1] Bioconductor
bit                    4.0.4    2020-08-04 [1] CRAN (R 4.0.2)
bit64                  4.0.5    2020-08-30 [1] CRAN (R 4.0.2)
bitops                 1.0-7    2021-04-24 [1] CRAN (R 4.0.3)
blob                   1.2.1    2020-01-20 [1] CRAN (R 4.0.2)
broom                  0.7.8    2021-06-24 [1] CRAN (R 4.0.3)
cachem                 1.0.5    2021-05-15 [1] CRAN (R 4.0.3)
Cairo                  1.5-12.2 2020-07-07 [1] CRAN (R 4.0.2)
cellranger             1.1.0    2016-07-27 [1] CRAN (R 4.0.2)
cli                    3.0.0    2021-06-30 [1] CRAN (R 4.0.3)
colorspace             2.0-2    2021-06-24 [1] CRAN (R 4.0.3)
crayon                 1.4.1    2021-02-08 [1] CRAN (R 4.0.3)
data.table           * 1.14.0   2021-02-21 [1] CRAN (R 4.0.3)
DBI                    1.1.1    2021-01-15 [1] CRAN (R 4.0.2)
dbplyr                 2.1.1    2021-04-06 [1] CRAN (R 4.0.3)
DelayedArray           0.16.3   2021-03-24 [1] Bioconductor
digest                 0.6.27   2020-10-24 [1] CRAN (R 4.0.2)
dplyr                * 1.0.7    2021-06-18 [1] CRAN (R 4.0.3)
edgeR                * 3.32.1   2021-01-14 [1] Bioconductor
ellipsis               0.3.2    2021-04-29 [1] CRAN (R 4.0.3)
evaluate               0.14     2019-05-28 [1] CRAN (R 4.0.2)
fansi                  0.5.0    2021-05-25 [1] CRAN (R 4.0.3)
fastmap                1.1.0    2021-01-25 [1] CRAN (R 4.0.2)
forcats              * 0.5.1    2021-01-27 [1] CRAN (R 4.0.2)
fs                     1.5.0    2020-07-31 [1] CRAN (R 4.0.2)
genefilter           * 1.72.1   2021-01-21 [1] Bioconductor
generics               0.1.0    2020-10-31 [1] CRAN (R 4.0.2)
GenomeInfoDb         * 1.26.7   2021-04-08 [1] Bioconductor
GenomeInfoDbData       1.2.4    2021-02-02 [1] Bioconductor
```

```
GenomicRanges        * 1.42.0   2020-10-27 [1] Bioconductor
ggplot2              * 3.3.5    2021-06-25 [1] CRAN (R 4.0.3)
glue                   1.4.2    2020-08-27 [1] CRAN (R 4.0.2)
gtable                 0.3.0    2019-03-25 [1] CRAN (R 4.0.2)
haven                  2.4.1    2021-04-23 [1] CRAN (R 4.0.3)
hms                    1.1.0    2021-05-17 [1] CRAN (R 4.0.3)
htmltools              0.5.1.1  2021-01-22 [1] CRAN (R 4.0.2)
httr                   1.4.2    2020-07-20 [1] CRAN (R 4.0.2)
IRanges              * 2.24.1   2020-12-12 [1] Bioconductor
IRdisplay              1.0      2021-01-20 [1] CRAN (R 4.0.2)
IRkernel               1.2      2021-05-11 [1] CRAN (R 4.0.3)
jsonlite               1.7.2    2020-12-09 [1] CRAN (R 4.0.2)
lattice                0.20-41  2020-04-02 [2] CRAN (R 4.0.3)
lifecycle              1.0.0    2021-02-15 [1] CRAN (R 4.0.3)
limma                * 3.46.0   2020-10-27 [1] Bioconductor
locfit                 1.5-9.4  2020-03-25 [1] CRAN (R 4.0.2)
lubridate              1.7.10   2021-02-26 [1] CRAN (R 4.0.3)
magrittr               2.0.1    2020-11-17 [1] CRAN (R 4.0.2)
Matrix                 1.3-4    2021-06-01 [1] CRAN (R 4.0.3)
MatrixGenerics       * 1.2.1    2021-01-30 [1] Bioconductor
matrixStats          * 0.59.0   2021-06-01 [1] CRAN (R 4.0.3)
memoise              * 2.0.0    2021-01-26 [1] CRAN (R 4.0.2)
mgcv                 * 1.8-33   2020-08-27 [2] CRAN (R 4.0.3)
modelr                 0.1.8    2020-05-19 [1] CRAN (R 4.0.2)
munsell                0.5.0    2018-06-12 [1] CRAN (R 4.0.2)
nlme                 * 3.1-152  2021-02-04 [1] CRAN (R 4.0.3)
pbdZMQ                 0.3-5    2021-02-10 [1] CRAN (R 4.0.3)
pillar                 1.6.1    2021-05-16 [1] CRAN (R 4.0.3)
pkgconfig              2.0.3    2019-09-22 [1] CRAN (R 4.0.2)
purrr                * 0.3.4    2020-04-17 [1] CRAN (R 4.0.2)
R6                     2.5.0    2020-10-28 [1] CRAN (R 4.0.2)
Rcpp                   1.0.7    2021-07-07 [1] CRAN (R 4.0.3)
RCurl                  1.98-1.3 2021-03-16 [1] CRAN (R 4.0.3)
readr                * 1.4.0    2020-10-05 [1] CRAN (R 4.0.2)
readxl                 1.3.1    2019-03-13 [1] CRAN (R 4.0.2)
repr                   1.1.3    2021-01-21 [1] CRAN (R 4.0.2)
reprex                 2.0.0    2021-04-02 [1] CRAN (R 4.0.3)
rlang                  0.4.11   2021-04-30 [1] CRAN (R 4.0.3)
RSQLite                2.2.7    2021-04-22 [1] CRAN (R 4.0.3)
rstudioapi             0.13     2020-11-12 [1] CRAN (R 4.0.2)
rvest                  1.0.0    2021-03-09 [1] CRAN (R 4.0.3)
S4Vectors            * 0.28.1   2020-12-09 [1] Bioconductor
scales                 1.1.1    2020-05-11 [1] CRAN (R 4.0.2)
sessioninfo            1.1.1    2018-11-05 [1] CRAN (R 4.0.2)
stringi                1.6.2    2021-05-17 [1] CRAN (R 4.0.3)
stringr              * 1.4.0    2019-02-10 [1] CRAN (R 4.0.2)
SummarizedExperiment * 1.20.0   2020-10-27 [1] Bioconductor
survival               3.2-7    2020-09-28 [2] CRAN (R 4.0.3)
```

```
 sva                 * 3.38.0   2020-10-27 [1] Bioconductor
 tibble              * 3.1.2    2021-05-16 [1] CRAN (R 4.0.3)
 tidyr               * 1.1.3    2021-03-03 [1] CRAN (R 4.0.3)
 tidyselect            1.1.1    2021-04-30 [1] CRAN (R 4.0.3)
 tidyverse           * 1.3.1    2021-04-15 [1] CRAN (R 4.0.3)
 utf8                  1.2.1    2021-03-12 [1] CRAN (R 4.0.3)
 uuid                  0.1-4    2020-02-26 [1] CRAN (R 4.0.2)
 vctrs                 0.3.8    2021-04-29 [1] CRAN (R 4.0.3)
 withr                 2.4.2    2021-04-18 [1] CRAN (R 4.0.3)
 XML                   3.99-0.6 2021-03-16 [1] CRAN (R 4.0.3)
 xml2                  1.3.2    2020-04-23 [1] CRAN (R 4.0.2)
 xtable                1.8-4    2019-04-21 [1] CRAN (R 4.0.2)
 XVector               0.30.0   2020-10-27 [1] Bioconductor
 zlibbioc              1.36.0   2020-10-27 [1] Bioconductor

[1] /home/jbenja13/R/x86_64-pc-linux-gnu-library/4.0
[2] /usr/lib/R/library
```