SIEVEseq Introduction

Hongxiang Li and Tsung Fei Khang

2026-06-13

Introduction

SIEVEseq is an R package for the simultaneous analysis of differential expression (DE), differential variability (DV), and differential skewness (DS) using RNA-Seq data. Unlike conventional RNA-Seq differential expression methods that focus primarily on differences in gene expression means, *SIEVEseq* jointly investigates changes in three characteristics of gene expression distributions: (i) Mean (DE); (ii) Variability (DV); (iii) Skewness (DS). The method adopts a compositional data analysis (CoDA) framework and models centered log-ratio (CLR) transformed RNA-Seq data using the skew-normal distribution. The location, scale, and skewness parameters of the skew-normal distribution are used to characterize expression mean, variability, and skewness, respectively. A unique feature of *SIEVEseq* is its ability to simultaneously detect genes exhibiting differential expression, differential variability, and differential skewness between two biological conditions. This enables a more comprehensive characterization of transcriptomic differences than methods focusing solely on differential expression. For a two-group comparison, *SIEVEseq* classifies genes according to all possible combinations of DE, DV, and DS patterns.

Installation

Install *SIEVEseq* from GitHub with

```
#remotes::install_github("Divo-Lee/SIEVEseq")
```

Getting Started

We start the R session with loading the *SIEVEseq* package as follows:

```
library(SIEVEseq)
```

We illustrate the analysis using a simulated dataset of CLR-transformed RNA-Seq counts, `clrCounts3`. This dataset contains 500 genes, with the first 50 genes exhibiting differential expression, and 200 samples per group (control vs. case). First, we load the simulated dataset `clrCounts3`.

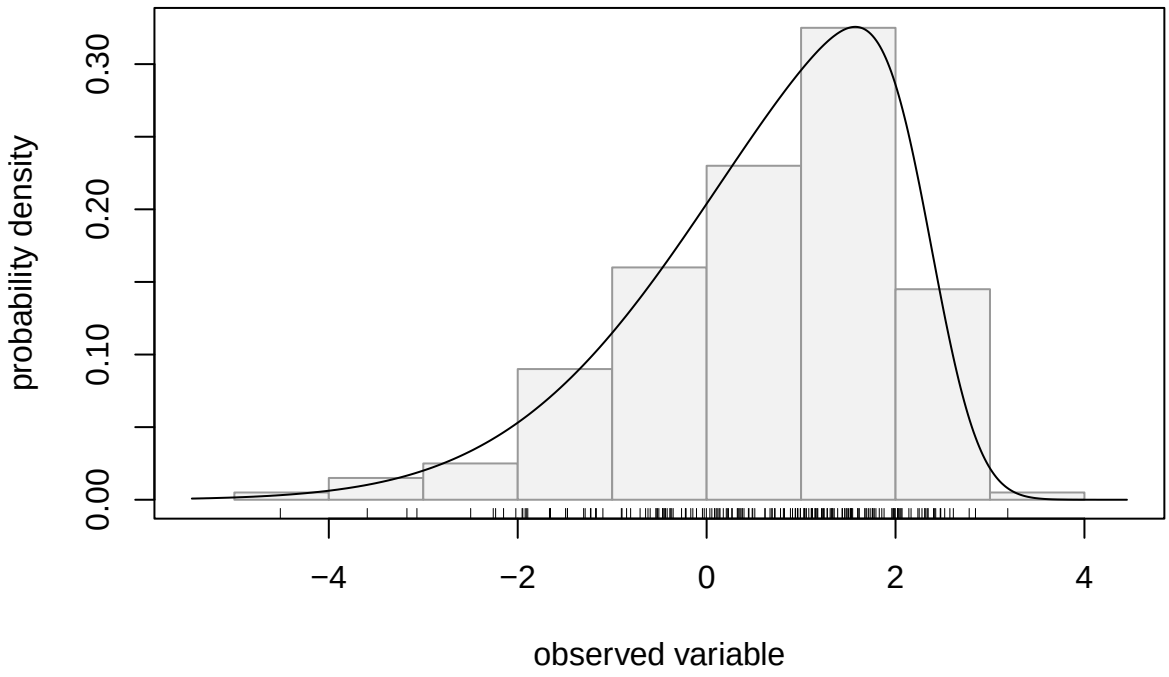
```
data("clrCounts3")
#500 genes, 200 samples per group, differential variability for the first 50 genes,
#CLR-transformed counts table
dim(clrCounts3)
#> [1] 500 400
clrCounts3[1:5, c(1:3, 201:203)]
#>          control1   control2   control3      case1      case2       case3
#> gene1 -4.7629127 -0.9996266 -3.0958239 -1.5056330 -0.7986745 -0.44705926
#> gene2  0.4880510 -1.1757562 -0.3877737  2.6615802  1.8591686 -0.02176843
#> gene3  0.4438375  1.6513383  0.2992945  1.2751293 -1.5370783 -1.25622425
```

```
#> gene4 -0.7375610 -3.3084422 -1.3505844  0.1098472 -0.2764101 -2.62677026
#> gene5  1.7766733 -0.2230978  0.8940016  3.8693180  3.5017365  2.89556528
```

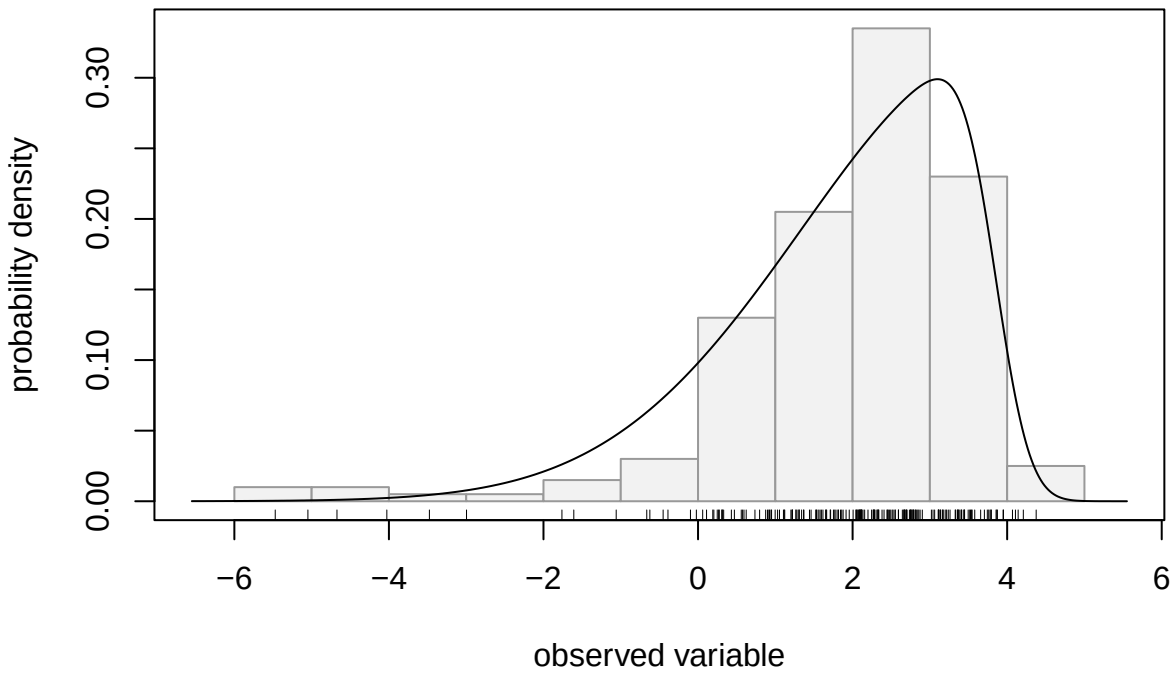
In the CLR-transformed count table, each row represents one gene, and each column represents one sample.

The function `SN.plot()` produces a histogram of observed CLR-transformed counts, with the fitted skew-normal probability density function for a particular gene/transcript. It can be used to graphically check the data fit skew-normal distribution well or not.

```
SN.plot(clrCounts3[2, 1:200]) # gene 2 in control group
```




```
SN.plot(clrCounts3[2, 201:400]) # gene 2 in case group
```



Above two Figures show that the skew-normal distribution fit the CLR-transformed counts of gene 2 in both control and case groups well.

The function `clr.SN.fit()` can be used to calculate the maximum likelihood estimate (MLE) of the mean (*mu*), scale (*sigma*, standard deviation) and skewness (*gamma*) parameters for genes (or a particular gene) of interest under one condition.

```
clr.SN.fit(clrCounts3[2, 1:200]) # gene 2 in control group
#>            mu         se.mu          z.mu          p.mu         sigma
#>  6.118498e-01  9.645466e-02  6.343393e+00  2.247594e-10  1.386965e+00
#>      se.sigma       z.sigma       p.sigma         gamma     se.gamma.
#>  7.791041e-02  1.780204e+01  6.813534e-71 -8.693680e-01  6.199292e-02
#>       z.gamma       p.gamma
#> -1.402367e+01  1.116928e-44
clr.SN.fit(clrCounts3[3:4, 201:400]) # gene 3 and gene 4 in case group
#>              mu      se.mu      z.mu         p.mu    sigma   se.sigma  z.sigma
#> gene3 -1.549623 0.10776916 -14.37909 7.000690e-47 1.541671 0.08530393 18.07268
#> gene4 -1.223296 0.09637255 -12.69341 6.432507e-37 1.371311 0.07634560 17.96189
#>            p.sigma      gamma  se.gamma.   z.gamma      p.gamma
#> gene3 5.230520e-73 -0.7965641 0.07476898 -10.65367 1.676229e-26
#> gene4 3.874054e-72 -0.7337640 0.10068099  -7.28801 3.145670e-13
```

Differential Exprssion, Variability and Skewness Analyses

`clrSeq()` is the function for estimating the mean, scale (standard deviation) and skewness parameters of the skew-normal distribution using CLR-transformed RNA-Seq data for two groups. The output of `clrSeq()`

will be used as the input of the function `clrSIEVE()`, which is the main function to run simultaneous DE, DV and DS tests between two conditions for RNA-Seq data. The output of `clrSIEVE()` is a list containing that containing four class objects: `clrDE_test`, `clrDV_test`, `clrDS_test` and `clrSIEVE_tests`. `clrDE_test`, `clrDV_test` and `clrDS_test` provide the results of DE, DV and DS tests, respectively. `clrSIEVE_tests` gives the results of all these three tests together.

Below is some examples showing how to use the output to perform the tests of DE, DV and DS.

Examples

We illustrate the DE analysis using simulated data, `clrCounts3`, which contains 200 samples per group with CLR-transformed counts of 500 genes (the first 50 genes are DE genes). We illustrate the DV analysis using simulated data, `clrCounts2`, which contains 200 samples per group with CLR-transformed counts of 500 genes (the first 50 genes are DV genes).

```
data("clrCounts3")
 #CLR-transformed counts table, 500 genes, 200 samples per group,
 #differential expression for the first 50 genes
data("clrCounts2")
 #CLR-transformed counts table, 500 genes, 200 samples per group,
 #differential variability for the first 50 genes,
dim(clrCounts3); dim(clrCounts2)
#> [1] 500 400
#> [1] 500 400
 #CLR-transformed counts table, 500 genes, 200 samples per group,
 #differential expression for the first 50 genes,
groups <- c(rep(0,200), rep(1,200))
 # 200 control samples, and 200 case samples
clrseq_result1 <-  clrSeq(clrCounts3, group = groups) # DE dataset
clrseq_result2 <-  clrSeq(clrCounts2, group = groups) # DV dataset
```

For DE, DV and DV tests, we are mainly interested in *mu1* and *mu2*, *sigma1* and *sigma2*, and *gamma1* and *gamma2*, respectively. The tests are based on the differences in each parameter between two groups.

```
head(clrseq_result1, 3) # MLE, DE genes
#>              mu1     se.mu1      z.mu1         p.mu1   sigma1  se.sigma1
#> gene1 -2.8184434 0.10596567 -26.597703 7.216311e-156 1.496367 0.08498745
#> gene2  0.6118498 0.09645466   6.343393  2.247594e-10 1.386965 0.07791041
#> gene3 -0.2319881 0.10474632  -2.214761  2.677648e-02 1.506399 0.08322000
#>       z.sigma1     p.sigma1     gamma1  se.gamma1  z.gamma1     p.gamma1
#> gene1 17.60691 2.180081e-69 -0.7077052 0.13108419  -5.39886 6.706575e-08
#> gene2 17.80204 6.813534e-71 -0.8693680 0.06199292 -14.02367 1.116928e-44
#> gene3 18.10141 3.106044e-73 -0.8791989 0.05209296 -16.87750 6.587633e-64
#>             mu2     se.mu2     z.mu2        p.mu2   sigma2  se.sigma2 z.sigma2
#> gene1 -1.543750 0.09976864 -15.47330 5.254117e-54 1.431188 0.07904078 18.10696
#> gene2  1.867091 0.10727165  17.40526 7.526465e-68 1.538065 0.08544785 18.00005
#> gene3 -1.549623 0.10776916 -14.37909 7.000690e-47 1.541671 0.08530393 18.07268
#>           p.sigma2     gamma2  se.gamma2  z.gamma2     p.gamma2
#> gene1 2.808175e-73 -0.8483146 0.06122393 -13.85593 1.171292e-43
#> gene2 1.946578e-72 -0.9192565 0.04484592 -20.49811 2.238222e-93
#> gene3 5.230520e-73 -0.7965641 0.07476898 -10.65367 1.676229e-26
 #
tail(clrseq_result1, 3) # MLE, non-DE genes
#>                mu1     se.mu1     z.mu1         p.mu1   sigma1  se.sigma1
#> gene498  1.5569868 0.09685569 16.075327  3.799860e-58 1.378766 0.07843358
#> gene499 -0.3298515 0.10250085 -3.218036  1.290715e-03 1.473067 0.08188897
```

```
#> gene500  2.4624770 0.08859045 27.796192 4.822728e-170 1.251526 0.06777246
#>          z.sigma1     p.sigma1     gamma1  se.gamma1   z.gamma1     p.gamma1
#> gene498 17.57877 3.582839e-69 -0.7997259 0.09449698  -8.462979 2.606333e-17
#> gene499 17.98858 2.393972e-72 -0.8643496 0.05996585 -14.414029 4.223351e-47
#> gene500 18.46659 3.835530e-76 -0.4831572 0.17231275  -2.803955 5.047993e-03
#>                mu2    se.mu2    z.mu2         p.mu2   sigma2  se.sigma2
#> gene498  1.8116923 0.0918193 19.73106  1.166924e-86 1.307257 0.07440820
#> gene499 -0.3011392 0.1001224 -3.00771  2.632240e-03 1.427857 0.07981323
#> gene500  2.4574969 0.1011258 24.30138 1.895696e-130 1.445073 0.08003977
#>          z.sigma2     p.sigma2     gamma2  se.gamma2   z.gamma2     p.gamma2
#> gene498 17.56872 4.276844e-69 -0.7808326 0.10402539  -7.506173 6.088082e-14
#> gene499 17.88997 1.411728e-71 -0.7673193 0.09100465  -8.431650 3.408264e-17
#> gene500 18.05444 7.279513e-73 -0.8157572 0.07137423 -11.429295 2.985224e-30
 #
```

```
head(clrseq_result2, 3) # MLE, DV genes
#>               mu1     se.mu1        z.mu1         p.mu1   sigma1  se.sigma1
#> gene1  2.31814892 0.09866529  23.49508127 4.579426e-122 1.397609 0.07936950
#> gene2 -0.00926001 0.10035246  -0.09227487  9.264797e-01 1.433909 0.08086669
#> gene3 -2.65804049 0.11833773 -22.46147946 9.884175e-112 1.702576 0.09392265
#>       z.sigma1     p.sigma1     gamma1  se.gamma1  z.gamma1     p.gamma1
#> gene1 17.60890 2.105079e-69 -0.7286988 0.11983462  -6.08087 1.195320e-09
#> gene2 17.73176 2.384478e-70 -0.8391264 0.07529653 -11.14429 7.635060e-29
#> gene3 18.12742 1.936243e-73 -0.8377466 0.06003965 -13.95322 3.007063e-44
#>               mu2     se.mu2      z.mu2         p.mu2    sigma2  se.sigma2
#> gene1  0.8368613 0.20395770   4.103112 4.076298e-05 2.9525593 0.16301208
#> gene2 -0.8489436 0.16584606  -5.118865 3.073799e-07 2.3846179 0.13377610
#> gene3 -1.8538850 0.04608987 -40.223261 0.000000e+00 0.6571823 0.03660058
#>       z.sigma2     p.sigma2     gamma2  se.gamma2   z.gamma2      p.gamma2
#> gene1 18.11252 2.538627e-73 -0.9322065 0.03564253 -26.154328 8.799177e-151
#> gene2 17.82544 4.485465e-71 -0.8844428 0.05637359 -15.688958  1.799864e-55
#> gene3 17.95551 4.345416e-72 -0.7461128 0.09639357  -7.740276  9.920145e-15
 #
tail(clrseq_result2, 3) # MLE, non-DV genes
#>                mu1     se.mu1      z.mu1          p.mu1   sigma1  se.sigma1
#> gene498 -3.6785756 0.10107125 -36.395864 4.949186e-290 1.447175 0.08103079
#> gene499 -0.4899749 0.10736567  -4.563609  5.028165e-06 1.550925 0.08596291
#> gene500  1.1554417 0.09822636  11.763051  6.050782e-32 1.394388 0.07722596
#>         z.sigma1     p.sigma1     gamma1  se.gamma1   z.gamma1     p.gamma1
#> gene498 17.85957 2.435258e-71 -0.7739435 0.08777467  -8.817390 1.171586e-18
#> gene499 18.04179 9.153225e-73 -0.9083204 0.04396621 -20.659511 8.017127e-95
#> gene500 18.05595 7.083310e-73 -0.7472706 0.09444397  -7.912317 2.526429e-15
#>                mu2     se.mu2      z.mu2          p.mu2   sigma2  se.sigma2
#> gene498 -3.5494339 0.09823905 -36.130579 7.510288e-286 1.411433 0.08082423
#> gene499 -0.5520555 0.10962205  -5.035989  4.753867e-07 1.558898 0.08998106
#> gene500  1.2260723 0.10255699  11.955034  6.110844e-33 1.465468 0.08005154
#>         z.sigma2     p.sigma2     gamma2  se.gamma2  z.gamma2     p.gamma2
#> gene498 17.46299 2.741804e-68 -0.9079291 0.05552450 -16.35186 4.218715e-60
#> gene499 17.32473 3.061299e-67 -0.9341624 0.04924363 -18.97022 3.006466e-80
#> gene500 18.30655 7.336865e-75 -0.7989872 0.06773451 -11.79587 4.099640e-32
 #
```

DE analysis

```
sieve_try1 <- clrSIEVE(clrSeq_result = clrseq_result1,
                       alpha_level = 0.05,
                       order_DE = FALSE,
                       order_LFC = FALSE,
                       order_DS = FALSE,
                       order_sieve = FALSE)
names(sieve_try1)
#> [1] "clrDE_test"     "clrDV_test"     "clrDS_test"     "clrSIEVE_tests"
```

```
DE_test_result1 <- sieve_try1$clrDE_test # results of DE tests
head(DE_test_result1, 3) # DE genes
#>              DE     se_DE      z_DE      pval_DE  adj_pval_DE        mu1
#> gene1  1.274693 0.1455421  8.758243 1.983177e-18 5.613071e-16 -2.8184434
#> gene2  1.255241 0.1442592  8.701291 3.281292e-18 8.572784e-16  0.6118498
#> gene3 -1.317635 0.1502863 -8.767494 1.826862e-18 5.613071e-16 -0.2319881
#>             mu2 de_indicator
#> gene1 -1.543750            1
#> gene2  1.867091            1
#> gene3 -1.549623            1
tail(DE_test_result1, 3) # non-DE genes
#>                   DE     se_DE        z_DE    pval_DE adj_pval_DE        mu1
#> gene498  0.254705466 0.1334609  1.90846529 0.05633111           1  1.5569868
#> gene499  0.028712294 0.1432861  0.20038430 0.84118004           1 -0.3298515
#> gene500 -0.004980159 0.1344422 -0.03704313 0.97045062           1  2.4624770
#>                mu2 de_indicator
#> gene498  1.8116923            0
#> gene499 -0.3011392            0
#> gene500  2.4574969            0
```

Genes with *adj_pval_de* < *alpha_level* were flagged as having differential expression. *DE* presents the mean difference between two groups, that is, *DE* = *mu2* - *mu1*. DE gene: *de_indicator* = 1; non-DE gene: *de_indicator* = 0.

DV analysis

```
sieve_try2 <- clrSIEVE(clrSeq_result = clrseq_result2,
                       alpha_level = 0.05,
                       order_DE = FALSE,
                       order_LFC = FALSE,
                       order_DS = FALSE,
                       order_sieve = FALSE)
names(sieve_try1)
#> [1] "clrDE_test"     "clrDV_test"     "clrDS_test"     "clrSIEVE_tests"

DV_test_result2 <- sieve_try2$clrDV_test
head(DV_test_result2, 3)
#>       SD_ratio        LFC         DV     se_DV       z_DV      pval_DV
#> gene1 2.112578  1.0790049  1.5549500 0.1813076   8.576308 9.796713e-18
#> gene2 1.663019  0.7338049  0.9507091 0.1563185   6.081873 1.187867e-09
#> gene3 0.385993 -1.3733533 -1.0453933 0.1008021 -10.370747 3.368805e-25
#>        adj_pval_DV   sigma1    sigma2 dv_indicator
#> gene1 1.751246e-15 1.397609 2.9525593            1
```

```
#> gene2 1.008621e-07 1.433909 2.3846179            1
#> gene3 5.720924e-22 1.702576 0.6571823            1
tail(DV_test_result2, 3)
#>          SD_ratio          LFC           DV     se_DV        z_DV   pval_DV
#> gene498 0.9753024 -0.036078523 -0.035741776 0.1144489 -0.31229469 0.7548166
#> gene499 1.0051407  0.007397482  0.007972857 0.1244436  0.06406803 0.9489161
#> gene500 1.0509757  0.071729299  0.071079891 0.1112299  0.63903563 0.5227998
#>         adj_pval_DV   sigma1   sigma2 dv_indicator
#> gene498           1 1.447175 1.411433            0
#> gene499           1 1.550925 1.558898            0
#> gene500           1 1.394388 1.465468            0
```

Genes with *adj_pval_dv* < *alpha_level* were flagged as having differential variability. *DV* presents the difference of the standard deviations between two groups, that is, *DV* = *sigma2* - *sigma1*. *LFC* presents the log fold change (LFC) for scale (standard deviation) parameters, that is, $LFC = log_2(sigma2/sigma1) = log_2 sigma2$ - $log_2 sigma1$. DV gene: *dv_indicator* = 1; non-Dv gene: *dv_indicator* = 0.

DS analysis

```
DS_test_result3 <- sieve_try2$clrDS_test
head(DS_test_result3, 3)
#>                DS      se_DS       z_DS   pval_DS adj_pval_DS     gamma1
#> gene1 -0.20350772 0.12502290 -1.6277635 0.1035750           1 -0.7286988
#> gene2 -0.04531641 0.09406141 -0.4817748 0.6299660           1 -0.8391264
#> gene3  0.09163386 0.11356267  0.8069012 0.4197234           1 -0.8377466
#>           gamma2 ds_indicator
#> gene1 -0.9322065            0
#> gene2 -0.8844428            0
#> gene3 -0.7461128            0
```

Genes with *adj_pval_ds* < *alpha_level* were flagged as having differential skewness. *DS* presents the skewness difference between two groups, that is, *DS* = *gamma2* - *gamma1*. DS gene: *ds_indicator* = 1; non-DS gene: *ds_indicator* = 0. Note that, to date RNA-Seq data simulator is not available for controlling the skewness pattern of gene expression. In real data analysis, the violin plots were used to visually check whether the computational results are correct or not, for the DS test.

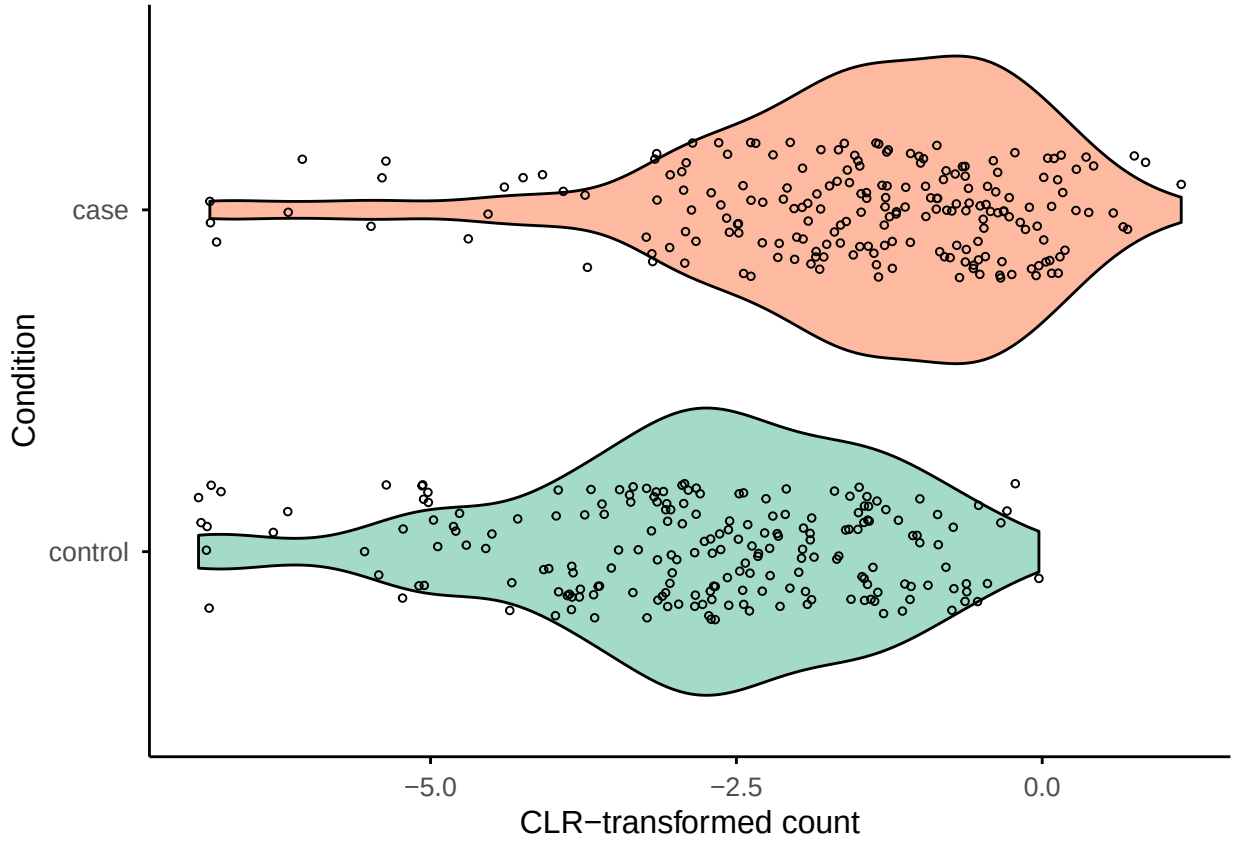
Simultaneous DE, DV and DS analysis

The results of above three tests (DE, DV and DS tests) can be provided by a class object `clrSIEVE_tests`, which contains the indicators for these three tests: `de_indicator`, `dv_indicator` and `ds_indicator`.

```
SIEVE_results <- sieve_try1$clrSIEVE_tests
head(SIEVE_results, 3)
#>              DE  adj_pval_DE  SD_ratio         LFC          DV adj_pval_DV
#> gene1  1.274693 5.613071e-16 0.9564423 -0.06425009 -0.06517823           1
#> gene2  1.255241 8.572784e-16 1.1089434  0.14918580  0.15110070           1
#> gene3 -1.317635 5.613071e-16 1.0234146  0.03339070  0.03527172           1
#>                DS adj_pval_DS de_indicator dv_indicator ds_indicator
#> gene1 -0.14060942           1            1            0            0
#> gene2 -0.04988853           1            1            0            0
#> gene3  0.08263478           1            1            0            0
```

The function `violin.plot.SIEVE()` produces violin plots for two groups comparing two groups DE, DV and DS, which is can be used graphically checking the computational results are correct or not.

```
violin.plot.SIEVE(data = clrCounts3,
                  "gene1",
                  group = groups,
                  group.names = c("control", "case")) # DE gene (non-DV and non-DS)
```




```
clrseq_result1[1,] # MLE, gene1 of clrCounts3. group 1: control; group 2: case
#>              mu1    se.mu1    z.mu1          p.mu1   sigma1  se.sigma1 z.sigma1
#> gene1 -2.818443 0.1059657 -26.5977 7.216311e-156 1.496367 0.08498745 17.60691
#>           p.sigma1     gamma1 se.gamma1 z.gamma1     p.gamma1      mu2
#> gene1 2.180081e-69 -0.7077052 0.1310842 -5.39886 6.706575e-08 -1.54375
#>            se.mu2    z.mu2        p.mu2   sigma2  se.sigma2 z.sigma2
#> gene1 0.09976864 -15.4733 5.254117e-54 1.431188 0.07904078 18.10696
#>           p.sigma2     gamma2  se.gamma2  z.gamma2     p.gamma2
#> gene1 2.808175e-73 -0.8483146 0.06122393 -13.85593 1.171292e-43
```

Notes on CLR-transfromation in *SIEVEseq*

Note that *SIEVEseq* does not perform CLR-transformation. CLR-transformed counts must be provided. Below is a simple example of CLR-transformation function for RNA-Seq count table:

```
#install.packages("compositions")
#library(compositions) # a package for compositional data analysis
# clr-transformation
clr.transform <- function(data = NULL){
  # data: count table, genes in rows and samples in columns
  data[data == 0] <- 1/2
```

```
  # A pseudo count 0.5 is added if the count is zero
  clr.count <- t(clr(t(data)))
  clr.count <- matrix(as.numeric(clr.count),
                      nrow = dim(data)[1],
                      ncol = dim(data)[2])
  row.names(clr.count) <- row.names(data)
  return(clr.count)
}
```