DASE user's guide

Xiang Liu xiang.liu@moffitt.org
Mingxiang Teng mingxiang.teng@moffitt.org
Department of Biostatistics and Bioinformatics, H. Lee Moffit Cancer Center & Research Institute, Tampa, FL, USA

2022-01-13

Contents

Introduction

Super enhancers (SEs) were proposed as broad regulatory domains on genome, usually spanning a minimum of thousands of base pairs and consisting of multiple constitute enhancers. The constitute enhancers work

together as a unit, instead of separately, to facilitate high enhancer activity. Aberrant SE activities, which are critical to understand disease mechanisms, could be raised by the alterations of one or more of their constitute enhancers. However, the state-of-art binary strategy in calling differential SEs only relies on overall activity changes, neglecting the local dynamics of constitute enhancers within SEs. DASE uses a weighted spline model to identify differential SEs from two conditions by accounting for the combinatorial effects of constitute enhancers weighted with their activities and locations (internal dynamics). In addition to overall changes, our medthod finds four novel types (*Shortened/lengthened*, *Shifted*, *Hollowed* and *other complex scenarios*) of differential SEs pointing to the structural differences within SEs.

Getting Started

Load the package in R

```
library(DASE)
```

Preparing Inputs

DASE requires the following input files:

1. enhancer bed file: a merged enhancer bed file includes the enhancer peaks of all samples.
2. SE bed file: a merged SE bed file includes the SE regions of all samples.
3. Coverage files: can either be the path of bam/bw files for each condition and replicate, or just an enhancer count table of all conditions and replicates.

Followings are the examples of enhancer and SE bed files

Enhancer bed file

The enhancer file can be any format of bed files, just make sure the first 6 columns contains the information of “chr”,“start”,“end”,“name”,“score”,and “strand”. Here is an example of bed files with only 6 columns.

```
# load enhancer bed file
enhancer_path <- system.file("extdata","enhancer.bed",package="DASE")
enhancer_region <- read.table(enhancer_path,sep="\t",header=F)
head(enhancer_region)
#>      V1       V2       V3         V4   V5 V6
#> 1 chr21 10119622 10119934 Peak_59320   58  .
#> 2 chr21 10413373 10414538  Peak_2651 1000  .
#> 3 chr21 13973708 13974647 Peak_51112   67  .
#> 4 chr21 14027434 14027662 Peak_40070   88  .
#> 5 chr21 14381282 14381485 Peak_44344   77  .
#> 6 chr21 14382640 14384785  Peak_2461 1000  .
```

SE bed file

The SE file can be any format of bed files, just make sure the first 6 columns contains the information of “chr”,“start”,“end”,“name”,“score”,and “strand”. Here is an example of bed files with only 6 columns.

```
# load SE bed file
se_path <- system.file("extdata","SE.bed",package="DASE")
se_region <- read.table(se_path,sep="\t",header=F)
head(se_region)
#>      V1       V2       V3                          V4   V5 V6
#> 1 chr21 21145883 21202220 17_Peak_26941_lociStitched  664  .
#> 2 chr21 43767083 43811262 10_Peak_18475_lociStitched 1080  .
#> 3 chr21 39347640 39389180 11_Peak_22013_lociStitched  887  .
#> 4 chr21 14737140 14764803  12_Peak_1337_lociStitched  847  .
#> 5 chr21 29634547 29660543 14_Peak_44019_lociStitched   81  .
#> 6 chr21 43880647 43903183  5_Peak_41657_lociStitched  949  .
```

Basic usage of *DASE*

In this section, we use DASE to find differential SEs with internal dynamics by comparing human chromosome 21 of two cancer cell lines (K562 and MCF7). Here, we focus on the basic usage of running DASE with different coverage input files (BAM, BigWig, and enhancer raw count table). This default setting will not include enhancer blacklist which indicates regions that don't have enhancers. In addition, under default setting, DASE will run permutation 10 times to get a significant threshold to identify SE categories. More additional features can be found later in the "Additional options" section.

Run DASE *with BAM or BigWig coverage files*

This section shows you how to use *DASE* with BAM or BigWig coverage files. The BAM or BigWig files are for the reads count of enhancers in different samples. *DASE* uses *featureCounts* to get the enhancer abundance of each sample with BAM file and *rtracklayer* with BigWig file. The first step is to get the path of BAM or BigWig file for each samples. Then with the imported enhancer and SE region files, we can run *DASE* as follow (Here we use BigWig files as example).

```
# path of BigWig file for each condition
s1_r1_bw <- system.file("extdata","K562_1_chr21.bw",package="DASE")
s1_r2_bw <- system.file("extdata","K562_2_chr21.bw",package="DASE")
s2_r1_bw <- system.file("extdata","MCF7_1_chr21.bw",package="DASE")
s2_r2_bw <- system.file("extdata","MCF7_2_chr21.bw",package="DASE")

# running DASE with BigWig files
DASE_out <- DASE(se_in=se_region,e_in=enhancer_region,data_type = "bw",
                 s1_r1_bam=s1_r1_bw,s1_r2_bam=s1_r2_bw,
                 s2_r1_bam=s2_r1_bw,s2_r2_bam=s2_r2_bw)
#> [1] "Step 1: merge and filter SE"
#> [1] "Step 2: calculate log2FC of constituent enhancer using Deseq2"
#> [1] "Step 3: b-spline fit log2FC"
#> [1] "Processing total of 34 SEs"
#> [1] "Step 4: permutation to get log2FC cutoff"
#> [1] "Permutation: 1"
#> [1] "Permutation: 2"
#> [1] "Permutation: 3"
#> [1] "Permutation: 4"
#> [1] "Permutation: 5"
#> [1] "Permutation: 6"
#> [1] "Permutation: 7"
```

```
#> [1] "Permutation: 8"
#> [1] "Permutation: 9"
#> [1] "Permutation: 10"
#> [1] "Step 5: pattern segments process"
#> [1] "Step 6: final category estimate"
```

Run DASE *with enhancer raw count table*

This section shows you how to use DASE with enhancer raw count table. The format of count table is shown bellow. Here we don't need BAM or BigWig files, because we already have the enhancer counts for each sample. We can run DASE as follow. This step will skip *featureCount*.

```
# read enhancer count table
enhancer_count_path <- system.file("extdata","chr21_enhancer_count.txt",package="DASE")
enhancer_count <- read.table(enhancer_count_path,sep="\t",header=T)
head(enhancer_count)
#>                 enhancer S1_r1 S1_r2 S2_r1 S2_r2
#> 1 chr21_5128185_5128529    12     2    21     5
#> 2 chr21_5240507_5241144    29    20    15     1
#> 3 chr21_5241953_5242568    33    29    11     2
#> 4 chr21_5242733_5243984   160    68    12     7
#> 5 chr21_5244027_5244554    52    29    13     4
#> 6 chr21_5244634_5245418    59    19    13     1

# run DASE
DASE_out_count <- DASE(se_in=se_region,e_in=enhancer_region,
                  enhancer_count_table=enhancer_count)
#> [1] "Step 1: merge and filter SE"
#> [1] "Step 2: calculate log2FC of constituent enhancer using Deseq2"
#> [1] "Step 3: b-spline fit log2FC"
#> [1] "Processing total of 34 SEs"
#> [1] "Step 4: permutation to get log2FC cutoff"
#> [1] "Permutation: 1"
#> [1] "Permutation: 2"
#> [1] "Permutation: 3"
#> [1] "Permutation: 4"
#> [1] "Permutation: 5"
#> [1] "Permutation: 6"
#> [1] "Permutation: 7"
#> [1] "Permutation: 8"
#> [1] "Permutation: 9"
#> [1] "Permutation: 10"
#> [1] "Step 5: pattern segments process"
#> [1] "Step 6: final category estimate"
```

Run DASE *with different spline functions*

DASE use spline functions to fit the log2 fold change value of constituent enhancers within each SEs.There are 3 spline functions (b-spline, natural spline, and smooth.spline) implemented in *DASE*. *DASE* use b-spline as default. Those 3 functions provide similar results, however they may have some different performance in some cases. The examples below shows some comparison of those spline functions.

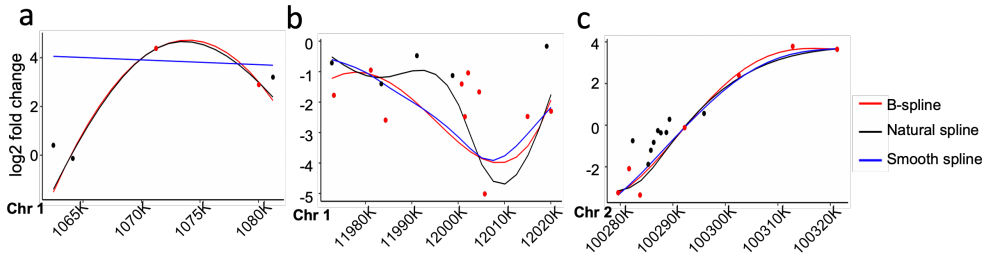


Figure 1: spline compare figuers.

Smooth spline can not handle small data points like others do (figure a). When there are large number of data points (usually larger than 6), all the spline functions has similar performance (figure c). However, in some cases, those spline functions may perform differently by how they handle the weights of each data point (figure b).

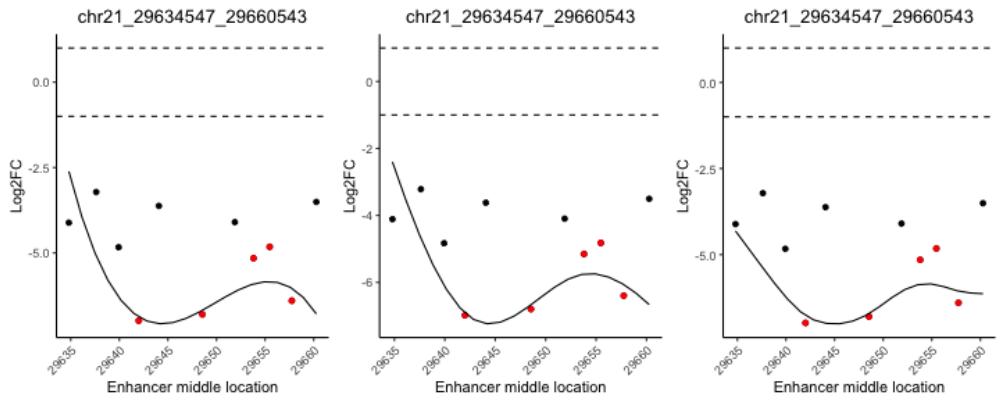
Users can chose their preferred spline function with *spline_fun* parameter. Following are some examples of run *DASE* with different spline functions.

```
# rund DASE with b-spline function (default)
DASE_out_bs <- DASE(se_in=se_region,e_in=enhancer_region,
                    enhancer_count_table=enhancer_count,permut=F)
#> [1] "Step 1: merge and filter SE"
#> [1] "Step 2: calculate log2FC of constituent enhancer using Deseq2"
#> [1] "Step 3: b-spline fit log2FC"
#> [1] "Processing total of 34 SEs"
#> [1] "Step 4: permutation to get log2FC cutoff"
#> [1] "Step 5: pattern segments process"
#> [1] "Step 6: final category estimate"

# run DASE with natural spline function
DASE_out_ns <- DASE(se_in=se_region,e_in=enhancer_region,
                    enhancer_count_table=enhancer_count,permut=F,
                    spline_fun = "ns")
#> [1] "Step 1: merge and filter SE"
#> [1] "Step 2: calculate log2FC of constituent enhancer using Deseq2"
#> [1] "Step 3: natural spline fit log2FC"
#> [1] "Processing total of 34 SEs"
#> [1] "Step 4: permutation to get log2FC cutoff"
#> [1] "Step 5: pattern segments process"
#> [1] "Step 6: final category estimate"

# run DASE with smooth spline function
DASE_out_smooth <- DASE(se_in=se_region,e_in=enhancer_region,
                    enhancer_count_table=enhancer_count,permut=F,
                    spline_fun = "smooth")
#> [1] "Step 1: merge and filter SE"
#> [1] "Step 2: calculate log2FC of constituent enhancer using Deseq2"
#> [1] "Step 3: smooth spline fit log2FC"
#> [1] "Processing total of 34 SEs"
#> [1] "Step 4: permutation to get log2FC cutoff"
#> [1] "Step 5: pattern segments process"
#> [1] "Step 6: final category estimate"
```

```
# spline fitted plots of different spline functions
library(gridExtra)
grid.arrange(DASE_out_bs$pattern_list[[7]],
             DASE_out_ns$pattern_list[[7]],
             DASE_out_smooth$pattern_list[[7]],nrow=1)
```



Interpretation of *DASE* output files

The output of *DASE* is a list with multiple data types including:

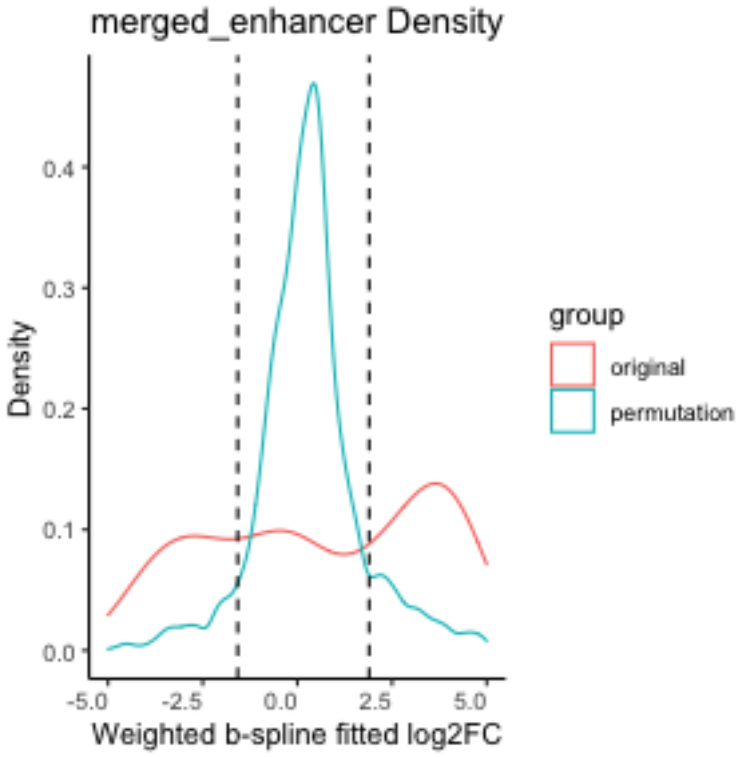
1. lfc_shrink: a shrinking lfc object from DESeq2, it can be used to get a MA plot
2. cutoff: significant threshold of fitted log2 fold change.
3. density_plot: a density plot of permutation and original fitted log2 fold change, if *permut=T*.
4. boxplot: a boxplot of final SE categories
5. se_category: a data frame contains final SE categories
6. pattern_list: a list contains figures of each SE's pattern
7. se_fit: a data frame contains DESeq2 output and spline-fitted log2 fold change of all constitute enhancers

Significant threshold

We use permutation of spline fitted log2 fold change to decide the significant threshold. Under default settings, *DASE* will run permutation 10 times with *SEpermut* function. If don't use permutation, the default significant threshold is -1 and 1. You can use your own threshold with *cutoff_v* parameter. Please refer to function manual or *Additional options* section.

```
# Significant threshold
DASE_out_count$cutoff
#> [1] -1.576203  1.894470
```

```
# Permutation density plot
DASE_out_count$density_plot
```



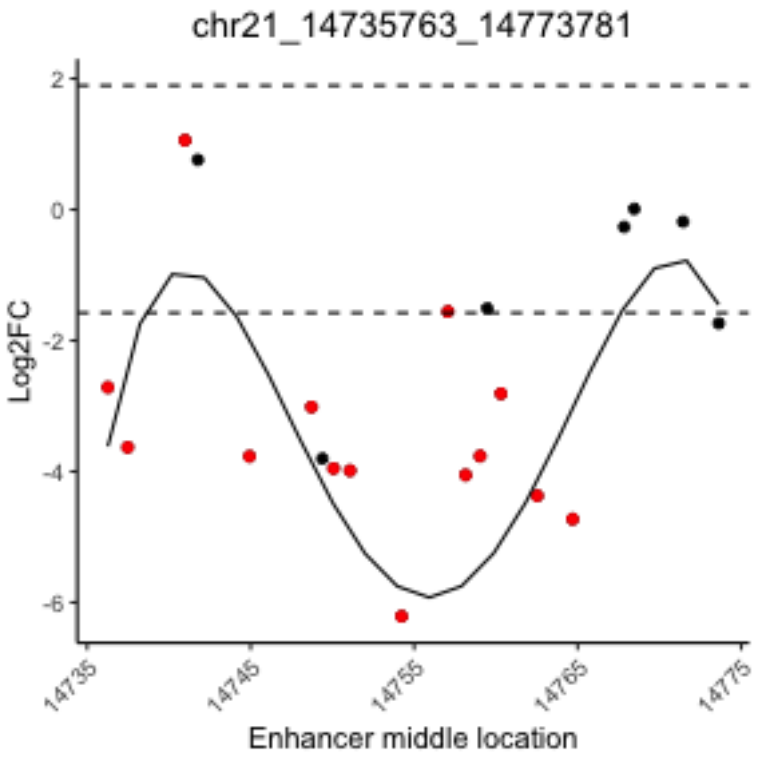
Black lines indicates the threshold which is obtained by the inflection point.

Super-enhancer internal dynamic categories

SE_category, pattern_list, se_fit are the output related to SE internal dynamics.

```
# se_categories
head(DASE_out_count$se_category)
#>              se_merge_name total_width number_enhancer category direction
#> 1  chr21_14735763_14773781    2415.972              21    Other      none
#> 61 chr21_45402658_45478681    1756.618              18    Other      none
#> 33 chr21_38784287_38854382    2417.733              21    Other      none
#> 54 chr21_43767083_43811262     816.333              13    Other      none
#> 43 chr21_40366721_40390218    6696.416               5  Overall         +
#> 12 chr21_29634547_29660543    4321.917              11  Overall         -
#>    non_mid_percent    mean_FC rank
#> 1            0.958 -2.5822940    1
#> 61           0.685  0.9187119    2
#> 33           0.510  1.1477844    3
#> 54           0.336  0.2621700    4
#> 43           1.000  5.1182229    1
#> 12           1.000 -4.8675004    2
```

```
# an example of one sample
DASE_out_count$pattern_list[[1]]
```



This figure shows the constitute enhancer patterns within SE "chr21_34863697_34890719" which identified as *Shortened*. Black line is the fitted log2 fold change curve, dots indicate constitute enhancers. Red dots indicate the constitute enhancers with large weights.

```
# example of se_fit
head(DASE_out_count$se_fit)
#>                e_merge_name   chr    start      end width S1_r1 S1_r2 S2_r1
#> 1: chr21_14751738_14756723 chr21 14751738 14756723  4986   463   471    48
#> 2: chr21_14760734_14764325 chr21 14760734 14764325  3592   254   303    91
#> 3: chr21_14750604_14751557 chr21 14750604 14751557   954    61    49    14
#> 4: chr21_14749603_14750540 chr21 14749603 14750540   938    52    46    14
#> 5: chr21_14757516_14758773 chr21 14757516 14758773  1258    63    33    11
#> 6: chr21_14737131_14737833 chr21 14737131 14737833   703    35    31    13
#>    S2_r2 S1_r1_norm S1_r2_norm S2_r1_norm S2_r2_norm  baseMean log2FoldChange
#> 1:    13  600.60927 1445.38402  11.401371  14.391655 517.94658      -6.333719
#> 2:    30  329.49191  929.83303  21.615100  33.211512 328.53789      -4.540646
#> 3:     9   79.12995  150.36904   3.325400   9.963453  60.69696      -4.262678
#> 4:     8   67.45504  141.16277   3.325400   8.856403  55.19990      -4.238023
#> 5:     7   81.72437  101.26894   2.612814   7.749353  48.33887      -4.337614
#> 6:     6   45.40243   95.13143   3.087871   6.642302  37.56601      -3.965940
#>        lfcSE       stat       pvalue         padj baseMean.1 log2FoldChange.1
#> 1: 0.6199495 -10.216508 1.672580e-24 2.263000e-22  517.94658        -6.211757
#> 2: 0.6328657  -7.174738 7.244567e-13 1.849415e-11  328.53789        -4.370066
#> 3: 0.7638403  -5.580588 2.397068e-08 2.206281e-07   60.69696        -3.986558
#> 4: 0.7751509  -5.467353 4.568069e-08 4.039606e-07   55.19990        -3.952824
#> 5: 0.7788941  -5.568940 2.562936e-08 2.327284e-07   48.33887        -4.053649
#> 6: 0.8182816  -4.846669 1.255515e-06 8.368035e-06   37.56601        -3.630087
#>      lfcSE.1     pvalue.1       padj.1           se_merge_name    s1_mean
#> 1: 0.6280041 1.672580e-24 2.263000e-22 chr21_14735763_14773781 1022.99665
```

```
#> 2: 0.6463209 7.244567e-13 1.849415e-11 chr21_14735763_14773781  629.66247
#> 3: 0.8052662 2.397068e-08 2.206281e-07 chr21_14735763_14773781  114.74949
#> 4: 0.8168441 4.568069e-08 4.039606e-07 chr21_14735763_14773781  104.30890
#> 5: 0.8223654 2.562936e-08 2.327284e-07 chr21_14735763_14773781   91.49666
#> 6: 0.8652202 1.255515e-06 8.368035e-06 chr21_14735763_14773781   70.26693
#>      s2_mean   max_mean width_mid   percent   cumsum spline_bs
#> 1: 12.896513 1022.99665  14754.23 42.343075 42.34308 -5.795081
#> 2: 27.413306  629.66247  14762.53 26.062496 68.40557 -4.140977
#> 3:  6.644427  114.74949  14751.08  4.749621 73.15519 -4.923321
#> 4:  6.090902  104.30890  14750.07  4.317472 77.47266 -4.501228
#> 5:  5.181083   91.49666  14758.15  3.787158 81.25982 -5.704292
#> 6:  4.865087   70.26693  14737.48  2.908434 84.16826 -2.311289
```

Additional options

In addition of default parameters, *DASE* can take an blacklist file which contains the regions cannot be identified as enhancers. *DASE* also can take a customized blacklist region with *custom_range* parameter. We have included a blacklist file from ENCODE (accession ID: ENCFF356LFX) in our package.

```
# blacklist file
blacklist_path <- system.file("extdata","region_blacklist.bed",package="DASE")
blacklist_region <- read.table(blacklist_path,sep="\t",header=F)
head(blacklist_region)
#>     V1       V2       V3
#> 1 chr1   628903   635104
#> 2 chr1  5850087  5850571
#> 3 chr1  8909610  8910014
#> 4 chr1  9574580  9574997
#> 5 chr1 32043823 32044203
#> 6 chr1 33818964 33819344
```

There is also an option whether to choice permutation or not. The default setting of DASE will run permutation 10 times. You can turn it off with *permut=F*. When there is no permutation, the default threshold is -1 and 1. You can chose any threshold you like with *cutoff_v* parameter. However, defining customized threshold is only available under *permut=F*.

DASE with enhancer blacklist region

Example of using blacklist options. Here, we focus on using DASE with enhancer count table.

```
# run DASE with blacklist file and customized region
DASE_out_bl <- DASE(se_in=se_region,e_in=enhancer_region,bl_file = blacklist_region,
                custom_range = c("chr21:14735763-29634779","chr21:33539902-43710703"),
                enhancer_count_table=enhancer_count)
#> [1] "Step 1: merge and filter SE"
#> [1] "Step 2: calculate log2FC of constituent enhancer using Deseq2"
#> [1] "Step 3: b-spline fit log2FC"
#> [1] "Processing total of 8 SEs"
#> [1] "Step 4: permutation to get log2FC cutoff"
#> [1] "Permutation: 1"
```

```
#> [1] "Permutation: 2"
#> [1] "Permutation: 3"
#> [1] "Permutation: 4"
#> [1] "Permutation: 5"
#> [1] "Permutation: 6"
#> [1] "Permutation: 7"
#> [1] "Permutation: 8"
#> [1] "Permutation: 9"
#> [1] "Permutation: 10"
#> [1] "Step 5: pattern segments process"
#> [1] "Step 6: final category estimate"
```

You can find that the number of SEs (8) in "DASE_out_bl" is less than "DASE_out_count" which is without blacklist region (34).

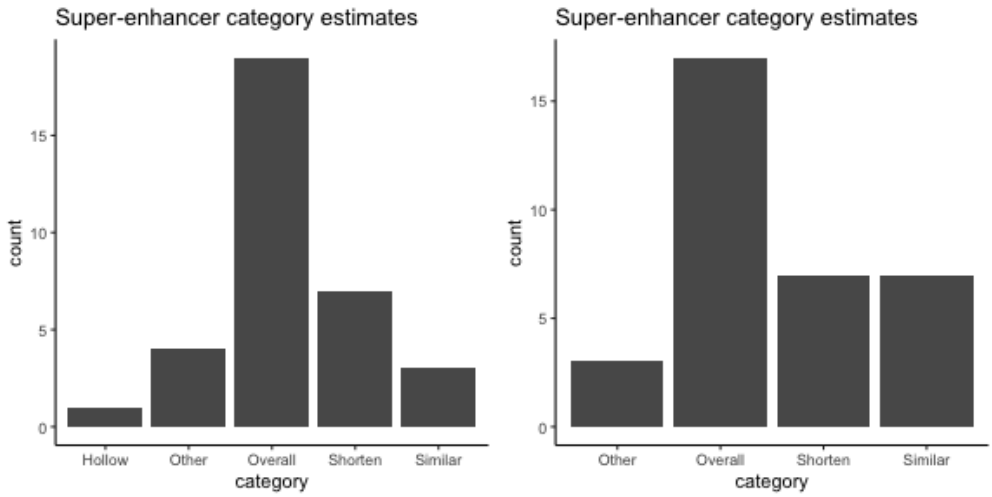
DASE with no permutation

Some examples of using permutation options. Here, we focus on using DASE with enhancer count table.

```
# run DASE with permutation 3 times
DASE_out <- DASE(se_in=se_region,e_in=enhancer_region,times=3,
                 cutoff_v = c(-2,2),enhancer_count_table=enhancer_count)
#> [1] "Step 1: merge and filter SE"
#> [1] "Step 2: calculate log2FC of constituent enhancer using Deseq2"
#> [1] "Step 3: b-spline fit log2FC"
#> [1] "Processing total of 34 SEs"
#> [1] "Step 4: permutation to get log2FC cutoff"
#> [1] "Permutation: 1"
#> [1] "Permutation: 2"
#> [1] "Permutation: 3"
#> [1] "Step 5: pattern segments process"
#> [1] "Step 6: final category estimate"
```

```
# run DASE with no permutation and customized threshold
DASE_out_p <- DASE(se_in=se_region,e_in=enhancer_region,permut = F,
                 cutoff_v = c(-3,3),enhancer_count_table=enhancer_count)
#> [1] "Step 1: merge and filter SE"
#> [1] "Step 2: calculate log2FC of constituent enhancer using Deseq2"
#> [1] "Step 3: b-spline fit log2FC"
#> [1] "Processing total of 34 SEs"
#> [1] "Step 4: permutation to get log2FC cutoff"
#> [1] "Step 5: pattern segments process"
#> [1] "Step 6: final category estimate"
```

```
# boxplots of different threshold
library(gridExtra)
grid.arrange(DASE_out_count$boxplot,
             DASE_out_p$boxplot,nrow=1)
```



Because we have a larger threshold (bottom plot), more SEs are identified as *similar* category than before (top plot).

Citation

If you used *DASE*, please cite our paper: [https://www.biorxiv.org/content/10.1101/2021.09.25.461810v1]