fdet: Find Differential Expressed Topologically Associating Domains (TADs)

This is a down-stream tool of Differential Gene Expression (DGE) analysis like DESeq2 or limma to find differential expressed TADs in genome. We cut each chromosome into 0.5Mb windows and access whether genes in each window were significantly one-side regulated (Figure1). This package could only be used on **human** data.

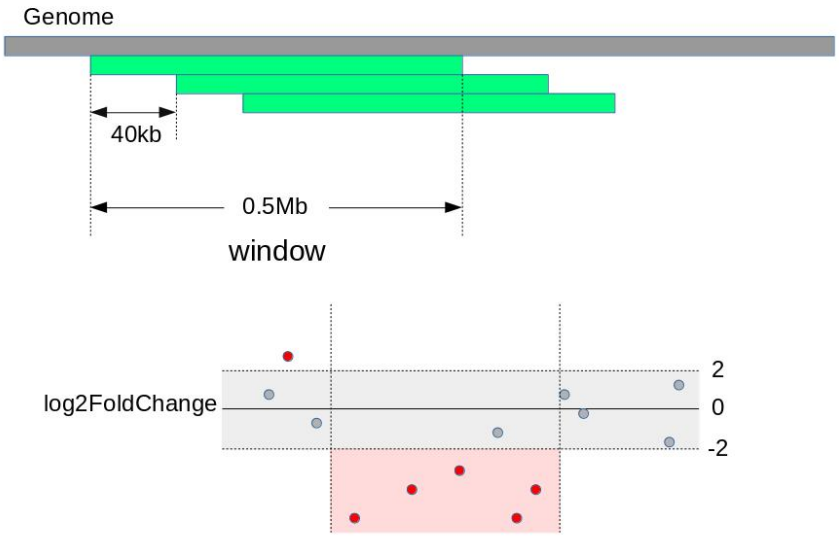


Figure 1 A work flow of the package.

Window regulation analysis:

Only windows with >= 5 genes and one-side regulated (all genes in the window were neutral/up-regulated or neutral/down-regulated) would be analyzed.
In each window log2FoldChange of each gene is $Gene_1$,$Gene_2$,...$Gene_i$
u is the mean of whole genome log2FoldChange and sd is the standard deviation of whole genome log2FoldChange.
One-sample wilcoxon test was used to evaluate log2FoldChange of the window.
H_0: median{$Gene_1$,$Gene_2$,...$Gene_i$} = u
H_1: median{$Gene_1$,$Gene_2$,...$Gene_i$} ≠ u

Z-score was also calculated as

$$Z\text{-}score = \sum_{i=1}^{n} \frac{Gene_i - u}{sd}$$

In the whole genome Z-score is a normal distribution(Figure 2). We filter outlier windows by Z-score of the window lies outside 95%CI.

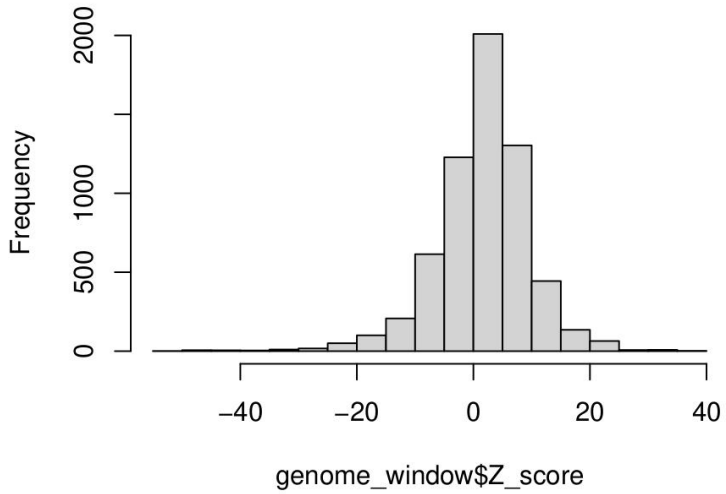


Figure 2 Window Z-score distribution in the whole genome.

In each window A Bayesian model was also used to evaluate log2FoldChange of the window. Genes with | log2FoldChange | > 2 were defined as $Gene_{diff}$. In each study, the expection rate of genes classified as $Gene_{diff}$ is calculated as

$$E = \frac{Count(Gene_{diff})}{Count(Gene_{all})}$$

Then in each window the expected number of genes classified as Genediff is

$$Count_{E\_w} = Count(Gene_{window}) \times E$$

The observed number of genes classified as Genediff is

$$Count_{O\_w}$$

Consider in each window the number of genes that were differentially expressed was a binomial distribution.

$$Count(Gene_{window\_diff}) \sim Bin(Count(Gene_{window}), E)$$

Then

$Prob(Count(Gene_{window\_diff}) = Count_{O\_w})$ was calculated by R function dbinom.

The criteria of TAD-wild regulation have not been defined. We set our filter criteria as follows:
1) Window size was 0.5Mb and bin size was 40kb.
2) Only windows with >= 5 genes and one-side regulated (all genes in the window were neutral/up-regulated or neutral/down-regulated) would be kept.
3) P-value of wilcox.test and Probability calculated by dbinom < 0.05.
4) Z-score of the window lies outside 95%CI of all windows in genome.
5) Differentially expressed genes observed was at least 2 more than expected.

Working pipeline

1. Install

Download fdep package by

```
git clone https://github.com/fancheyu5/fdet.git
```

R package devtools and ggplot2 should be pre-installed. If not, use

```
install.packages("devtools")
install.packages("ggplot2")
```

Install the package by

```
devtools::install_local('fdet_0.1.1.tar.gz') #path to fdet_0.1.1.tar.gz
```

2.Prepare input data

This is a down-stream tool of Differential Gene Expression (DGE) analysis like DESeq2 or limma. Surpose DGEresult is the data.frame result of DESeq2, then it should be like this:

```
> head(DGEresult)
                    HepG2   HepG2.215   baseMean log2FoldChange     lfcSE
ENSG00000160072 334.507825 539.0408623 364.598631     2.28325053 0.2216685
ENSG00000142611   0.000000   0.5074672   4.117934    -6.83100140 2.2645912
ENSG00000157911 390.725385 303.0873825 306.821941     0.39729924 0.1976312
ENSG00000269896   6.766354  14.5033745   9.312283    -0.79557349 0.8694569
ENSG00000228463   0.000000   1.0613698   1.295909     0.09314705 2.5176004
                      stat       pvalue         padj
ENSG00000160072 10.30029159 7.024804e-25 1.017033e-23
ENSG00000142611 -3.01643908 2.557626e-03 6.811494e-03
ENSG00000157911  2.01030616 4.439880e-02 9.442970e-02
ENSG00000269896 -0.91502345 3.601793e-01 5.748025e-01
ENSG00000228463  0.03699835 9.704863e-01           NA
```

HepG2 and HepG2.215 is the two groups in this demo.

Next annotate this data.frame with genome location. Users could use their own annotation pipeline. Just make sure the annotated data.frame has the following columns :

log2FoldChange chr start end gene_name

*chromosome number in “chr” column should not contain string “chr”

*gene_name is the label plotted in figure.

We put a hg38.gene.bed generated from Homo_sapiens.GRCh38.109.gtf in the demo file. Users could use it by:

```
gene.names <- read.table(file="demo/hg38.gene.bed",header=F,sep="\t",stringsAsFactors=F)
colnames(gene.names) <-    c("chr","start","end","gene_ID","gene_name")
library(dplyr) #if not installed, use        install.packages("dplyr")
DGEresult$gene_ID <- row.names(DGEresult)
DGE <- left_join(DGEresult,gene.names,by= "gene_ID")
```

```
> head(DGE)
       HepG2   HepG2.215   baseMean log2FoldChange     lfcSE        stat
1 334.507825 539.0408623 364.598631     2.28325053 0.2216685 10.30029159
2   0.000000   0.5074672   4.117934    -6.83100140 2.2645912 -3.01643908
3 390.725385 303.0873825 306.821941     0.39729924 0.1976312  2.01030616
4   6.766354  14.5033745   9.312283    -0.79557349 0.8694569 -0.91502345
5   0.000000   1.0613698   1.295909     0.09314705 2.5176004  0.03699835
        pvalue         padj         gene_ID  chr   start     end gene_name
1 7.024804e-25 1.017033e-23 ENSG00000160072    1 1471765 1497848    ATAD3B
2 2.557626e-03 6.811494e-03 ENSG00000142611    1 3069168 3438621    PRDM16
3 4.439880e-02 9.442970e-02 ENSG00000157911    1 2403964 2413797     PEX10
4 3.601793e-01 5.748025e-01 ENSG00000269896 <NA>      NA       NA      <NA>
5 9.704863e-01           NA ENSG00000228463 <NA>      NA       NA      <NA>
```

*Only protein coding genes were included in demo/hg38.gene.bed file.

To run the demo:

```
DGEresult <-    read.table(file="demo/RNAdiff.tsv",header=T,sep="\t",stringsAsFactors=F)
gene.names <- read.table(file="demo/hg38.gene.bed",header=F,sep="\t",stringsAsFactors=F)
colnames(gene.names) <-    c("chr","start","end","gene_ID","gene_name")
library(dplyr)        #if not installed, use        install.packages("dplyr")
DGE <- left_join(DGEresult,gene.names,by= "gene_ID")
```

3. Find differential expressed TADs

```
library(fdet)
res <- search_genome(DGE)
```

The *res* has the following columns:

	A	B	C	D	E	F	G	H	I	J	K	L
1	windowNo	window_5pos	window_3pos	gene_number	E_w	O_w	mean_log2FC	P.wilcox	regulation	Z_score	delta_g	P.binom
2	chr1_907	36240000	36740000	6	1.499748	4	3.1423489203	0.03125	UPregulated	40.6742	2.5003	0.03294
3	chr1_908	36280000	36780000	6	1.499748	4	3.1423489203	0.03125	UPregulated	40.6742	2.5003	0.03294
4	chr1_1166	46600000	47100000	7	1.749706	5	-5.403201585	0.03125	DOWNregulated	-55.7415	3.2503	0.01153
5	chr1_5079	203120000	203620000	6	1.499748	4	-5.855850788	0.03125	DOWNregulated	-60.8485	2.5003	0.03294
6	chr1_5080	203160000	203660000	7	1.749706	5	-5.488788644	0.015625	DOWNregulated	-56.7071	3.2503	0.01153

WindowNo	Window ID. Could be used in *plot_window* and *plot_window_with_TADs* functions.
window_5pos	5' posision of the window.
window_3pos	3' posision of the window.
gene_number	Count of genes in this window.
E_w	Expected count of | log2FoldChange |>2 genes in this window.
O_w	Observed count of | log2FoldChange |>2 genes in this window.
mean_log2FC	Mean of gene log2FoldChange in this window.
P.wilcox	P-value of median log2FoldChange ≠ 0 conducted by *wilcox.test* funtion.
Regulation	Direction of differential expression of the window.
Z_score	Z-score of the window.
delta_g	O_w – E_w
P.binom	Probability of Count(Gene$_{diff}$)=O_w conducted by *dbinom* funtion.

In this demo a total of 42 windows remained after filter.

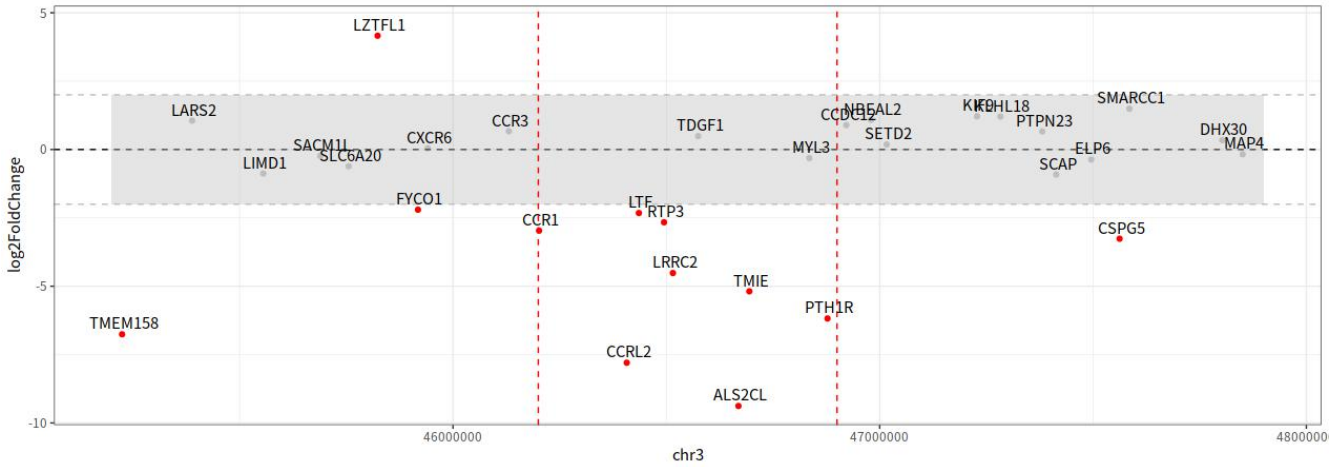
4. Plot differential expressed windows

Function *plot_window* takes three prameters:

DGE	– data.frame	The annotated DGE result with genome location.
Res	– data.frame	The data.frame generated by function *search_genome*
windowID	– vector of string	Just use the first and last windowID for continua windows.

To run demo:

```
options(scipen=200)# do not use scientific notation in xlab
plot_window(DGE,res,c("chr3_1156","chr3_1161"))
```



5. Plot differential expressed windows with TADs

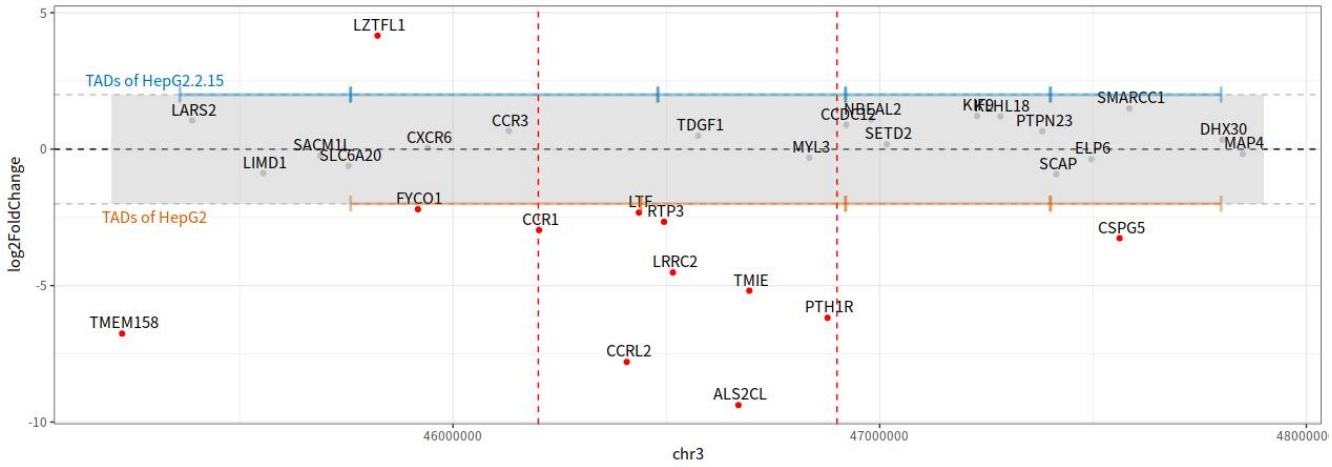
Function *plot_window_with_TADs* could plot window with TADs information with following parameters:

DGE	– data.frame	Same as	*plot_window*
Res	– data.frame	Same as	*plot_window*
TADs1	– data.frame	A data.frame with group1 TADs.bed.	Colnames should be chr start end
TADs2	– data.frame	A data.frame with group2 TADs.bed.	Colnames should be chr start end
group1	– string	name of group1	
group2	– string	name of group2	
windowID	– vector of string	Same as	*plot_window*

* chromosome name in tads.bed should not contain string 'chr'

To run demo:

```
TADs1=read.csv(file='demo/tad1.bed',sep='\t',header=F,stringsAsFactors=F)
colnames(TADs1)[1:3]<-c("chr","start","end")
TADs2=read.csv(file='demo/tad2.bed',sep='\t',header=F,stringsAsFactors=F)
colnames(TADs2)[1:3]<-c("chr","start","end")
plot_window_with_TADs(DGE,res,TADs1,TADs2,"HepG2.2.15","HepG2",c("chr3_1156","chr3_1161"))
```



The red dash lines marked target windows.