Multiple Beta t-Tests of Differential Transcription of mRNAs Between Two Conditions with Small Samples

Yuan-De Tan
tanyuande@gmail.com

May 3, 2016

Abstract

A major task in the analysis of count data of RNA reads from RNA-Seq is the detection of differentially expressed genes or isoforms. The count data are presented as a matrix consisting of RNA isoform annotation and the number of reads. Analogous analyses also arise for other assay types, such as comparative ChIP-Seq. The package *MBttest* provides a powerful method to test for differential expression by use of the beta distribution and gene- or isoform-specific variable ρ to control fudge effect due to small sample size. [1]. This vignette explains the use of the package. For more detail of the statistical method, please see our paper [**?**].

Contents

1 Introduction

This vignette is intended to give a rapid introduction to the commands used in implementing new beta t-test methods of evaluating differential expression in high-throughput sequencing data by means of the *MBttest***R** package. For fuller details on the methods being used, consult Tan et al (2015) [**?**] .

We assume that we have count data from a set of sequencing or other high-throughput experiments, arranged in an array such that except gene annotation information and id, each column describes a library and each row describes RNA tag or isoform for which data have been acquired. For example, the rows may correspond to the different sequences observed in a sequencing experiment. The data then consists of the number of each sequence observed in each sample. We wish to determine which, if any, rows of the data correspond to some patterns of differential expression across the samples.

MBttest, also called *mBetat* test, uses new beta t-test method to identify differential expression for each row. This approach introduces a gene- or isoform-specific variable, called ρ, into t-statistics to control fudge effect resulted from small samples. It has higher work efficiency than existing methods for identifying differential expressions of genes or isoforms either by inflating t-values with

$$\rho > \omega$$

threshold or by shrinking those with

$$\rho < \omega$$

[**?**] when number of replicate libraries in each condition is small, for example, equal to or less than 8.

Different from the exiting methods such as *baySeq* [**?**], *edgeR* Exact test [**?**] and[**?**], *edgeR GLM* [**?**] and [**?**], *DESeq* [**?**] and*DESeq2* [**?**], etc, *MBttest* requires performance of simulation to determine *threshold* ω before running program mbetattest. *MBttest* provides negative binomial simulation program to generate null count data without inputting arguments. User should repeat five or more simulations, perform program smbettest to produce null results and calculate ω using the method given in our paper [**?**].

[1] Other Bioconductor packages with similar aims are *edgeR*, *baySeq*, *DESeq* and *DESeq2*.

2 Data Preparation and Input

We begin by loading the *MBttest* package.

```
> library(MBttest)
```

MBttest requires data file contain two parts: Annotation information and count data. Information consists of `tagid`, `geneid`, gene name, chromosome `id`, DNA strand, etc. Information is in left side. It has at least one column for `geneid` or `tagid (isoformid)`. The data contain two conditions each having several replicate libraries and must be in right side. Here is an example :

```
> data(jkttcell)
> jkttcell[1:10,]
```

```
   tagid geneid    name   chr strand      pos  anno Jurk.NS.A Jurk.NS.B Jurk.NS.C
1     54  58998    COMT chr22      + 19956542    sg     66.80     43.48      4.65
2    111  59029    CRKL chr22      + 21308033 tu-ce     68.75     63.94     66.46
3    171  59104 SLC2A11 chr22      + 24227723    sg      2.86      2.67      8.15
4    231  59157  ADRBK2 chr22      + 26118985    tu     12.88      8.45      8.59
5    242  59164    SRRD chr22      + 26887904    sg     62.54     59.88     83.27
6    265  59184    HSCB chr22      + 29153206    tu      4.02      2.07      6.85
7    306  59209    UCRC chr22      + 30165939    sg    516.71    594.71     83.84
8    327  59212   MTMR3 chr22      + 30426495    tu      4.08      3.99      2.93
9    445  59310    NCF4 chr22      + 37274056    sg      0.00      1.30      0.03
10   472  59321   CYTH4 chr22      + 37711384    sg      4.40      1.60      0.12
   Jurk.48h.A Jurk.48h.B Jurk.48h.C
1       32.99      25.49      14.68
2       80.42      63.89      72.48
3       12.95      12.70       8.81
4       13.13      35.34       9.78
5       54.99      51.15      66.61
6        9.66       4.02       5.87
7      254.81     142.26     156.72
8       10.38      14.23       6.85
9       71.26      25.89      11.73
10      37.61      18.54      16.64
```

User also may use `head` to display the data with top 6 lines:

```
> head(jkttcell)
```

```
  tagid geneid    name   chr strand      pos  anno Jurk.NS.A Jurk.NS.B Jurk.NS.C
1    54  58998    COMT chr22      + 19956542    sg     66.80     43.48      4.65
2   111  59029    CRKL chr22      + 21308033 tu-ce     68.75     63.94     66.46
3   171  59104 SLC2A11 chr22      + 24227723    sg      2.86      2.67      8.15
4   231  59157  ADRBK2 chr22      + 26118985    tu     12.88      8.45      8.59
5   242  59164    SRRD chr22      + 26887904    sg     62.54     59.88     83.27
6   265  59184    HSCB chr22      + 29153206    tu      4.02      2.07      6.85
  Jurk.48h.A Jurk.48h.B Jurk.48h.C
1      32.99      25.49      14.68
2      80.42      63.89      72.48
3      12.95      12.70       8.81
4      13.13      35.34       9.78
5      54.99      51.15      66.61
6       9.66       4.02       5.87
```

If the data are *csv* file, user can use `read.csv` function to input data into **R Console** or If the data are *txt* file, user can use `read.delim` function to load data into **R Console**. After loading data, user should check the data inputted. `jkttcell` shows an example. In this example, 7 columns in the left side are information of *poly(A)* sites. The count data are listed in the right side.

3 Simulation for Calculating $omega$ Value

Before performing `mbetattest` on the real data, user needs simulation to determine $omega$ value. There are three steps for doing so:

3.1 Step1: Simulate null count data

Use the following function to generate null simulation data

`simulat(yy, nci, r1, r2, p, q, A)`

where
yy is real data.
$r1$ and $r2$ are replicate numbers in conditions 1 and 2.
p is proportion of genes differentially expressed in m genes, default value is 0.
q is proportion of genes artificial noise. Its default value is 0.
A is effect value. Its default value is 0. *nci*: column number of information of data.
Here is an example:

```
> sjknull1<-simulat(yy=jkttcell[1:500,],nci=7,r1=3,r2=3,q=0.2)
```

Our data is *jkttcell*. It has 7 columns for information of *poly(A)* sites. Two conditions are resting and stimulation. Each has 3 replicate libraries, r1=3 and r2=3. Since this is null simulation, we set p=0 and q=0.2 for artificial noise. With the same read data and parameters, you can generate a set of 4-6 null datasets: *sjknull2*, $\cdots$, *sjknull6* for calculating $omega$ value.

3.2 Step2: Perform multiple beta t-tests

Use function `smbetattest` to perform multiple beta t-test with ρ =1 on the simulated null data:

`smbetattest`(X, na, nb,alpha)

where
X=simulated data.
na and nb are numbers of replicate libraries in conditions 1 and 2.
alpha is probabilistic threshold. User can set *alpha*=0.05 or 0.01.
The example is

```
> mysim1<-smbetattest(X=sjknull1,na=3,nb=3,alpha=0.05)
```

Save them to *csv* files

`write.csv`*(mysim1, file='simulatedNullData1Result.csv")*

After performing `smbetattest` on each simulated null dataset, user would have results recorded in file like *simulatedNullData1Result.csv* and open it with excel.

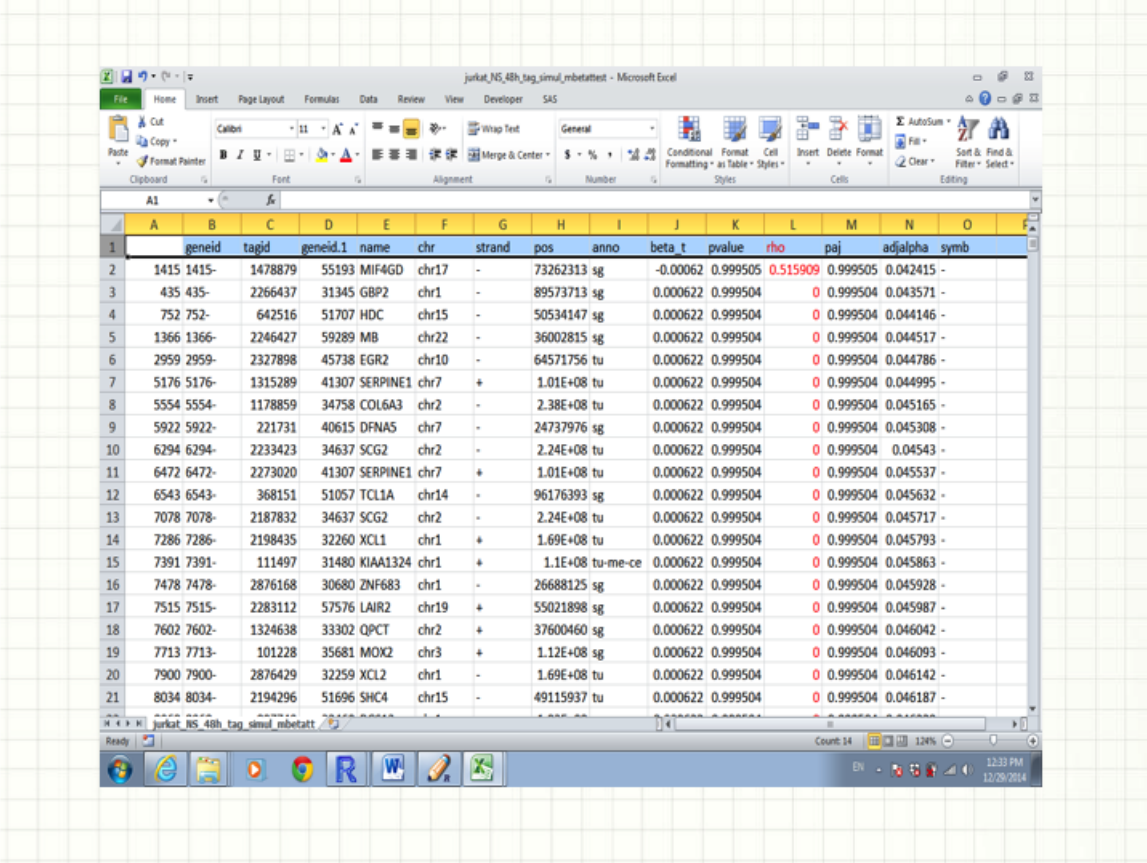
In symbol column, `mbeta` t-test gives test result: *symb*= "-" means that the gene or tag is not chosen while *symb*= "+" indicates that the gene or isoform is found to be differentially expressed.

In this example, 12 genes would be found to be falsely positive.

3.3 Step3: Calculate $omega$ value

Here is a demo for calculating $omega$(since we can't use greek letter $omega$ in R function, we use W to represent $omega$). In Figure 3, red highlighted column is *rho* column. We copied the ρ values of these 12 genes into another empty column and sorted them from the smallest to the largest. Then we gave sequence numbers from 1 to 12 corresponding to ρ-values and calculate q-value for each ordered ρ value:

We chose the 10th ρ value (1.09166) as the first W value because the 11th rho value has q-value $>$ 0.85. Repeat this process in 4-6 simulated null datasets and we took the averaged W value as W value in the real data.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1		geneid	tagid	geneid.1	name	chr	strand	pos	anno	beta_t	pvalue	rho	paj	adjalpha	symb
2	1415	1415-	1478879	55193	MIF4GD	chr17	-	73262313	sg	-0.00062	0.999505	0.515909	0.999505	0.042415	-
3	435	435-	2266437	31345	GBP2	chr1	-	89573713	sg	0.000622	0.999504	0	0.999504	0.043571	-
4	752	752-	642516	51707	HDC	chr15	-	50534147	sg	0.000622	0.999504	0	0.999504	0.044146	-
5	1366	1366-	2246427	59289	MB	chr22	-	36002815	sg	0.000622	0.999504	0	0.999504	0.044517	-
6	2959	2959-	2327898	45738	EGR2	chr10	-	64571756	tu	0.000622	0.999504	0	0.999504	0.044786	-
7	5176	5176-	1315289	41307	SERPINE1	chr7	+	1.01E+08	tu	0.000622	0.999504	0	0.999504	0.044995	-
8	5554	5554-	1178859	34758	COL6A3	chr2	-	2.38E+08	tu	0.000622	0.999504	0	0.999504	0.045165	-
9	5922	5922-	221731	40615	DFNA5	chr7	-	24737976	sg	0.000622	0.999504	0	0.999504	0.045308	-
10	6294	6294-	2233423	34637	SCG2	chr2	-	2.24E+08	tu	0.000622	0.999504	0	0.999504	0.04543	-
11	6472	6472-	2273020	41307	SERPINE1	chr7	+	1.01E+08	tu	0.000622	0.999504	0	0.999504	0.045537	-
12	6543	6543-	368151	51057	TCL1A	chr14	-	96176393	sg	0.000622	0.999504	0	0.999504	0.045632	-
13	7078	7078-	2187832	34637	SCG2	chr2	-	2.24E+08	tu	0.000622	0.999504	0	0.999504	0.045717	-
14	7286	7286-	2198435	32260	XCL1	chr1	+	1.69E+08	tu	0.000622	0.999504	0	0.999504	0.045793	-
15	7391	7391-	111497	31480	KIAA1324	chr1	+	1.1E+08	tu-me-ce	0.000622	0.999504	0	0.999504	0.045863	-
16	7478	7478-	2876168	30680	ZNF683	chr1	-	26688125	sg	0.000622	0.999504	0	0.999504	0.045928	-
17	7515	7515-	2283112	57576	LAIR2	chr19	+	55021898	sg	0.000622	0.999504	0	0.999504	0.045987	-
18	7602	7602-	1324638	33302	QPCT	chr2	+	37600460	sg	0.000622	0.999504	0	0.999504	0.046042	-
19	7713	7713-	101228	35681	MOX2	chr3	+	1.12E+08	sg	0.000622	0.999504	0	0.999504	0.046093	-
20	7900	7900-	2876429	32259	XCL2	chr1	-	1.69E+08	tu	0.000622	0.999504	0	0.999504	0.046142	-
21	8034	8034-	2194296	51696	SHC4	chr15	-	49115937	tu	0.000622	0.999504	0	0.999504	0.046187	-

Figure 1: This is an example showing the results obtained by `smbetattest` from simulated null data. Column A is row number, `geneid` is set in simulation, and `geneid.1` is original `geneid`.

4 Normalize the count data

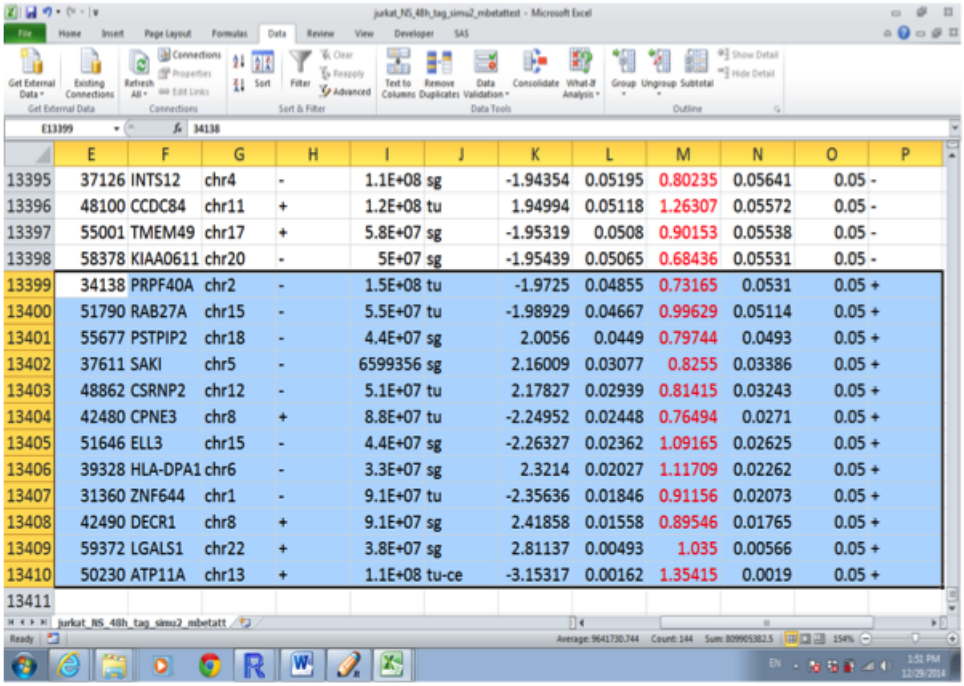
As a second processing step, we need to estimate the effective library size. This step is also called *normalization*, even though it may not make the count data be of normal distribution. If the counts of expressed genes in one condition are, on average, twice as high as in another (because the library was sequenced twice as deeply), the size factor for the first condition should be twice higher than the second one, then differential analysis would give error results. For this reason, we must make all libraries have the same size before performing any statistical method. For doing so, user can the function `estimateSizeFactors` of **R** package *DESeq* [**?**] or *DESeq2* [**?**] to estimate the size factors from the count data or use the following simple method to normalize the the count data: In excel sheet, use function sum to calculate sizes of all libraries, and then use excel function average to calculate averaged library size. The last step is to use the following equation to convert the original count data to new count data with the same library size:

$$Y_{ik} = \frac{y_{ij}\bar{N}}{N_j}$$

where $i = 2, \cdots, n$(number of genes or isofoms) in rows in a sheet, $j = nci + 1, \cdots, nci + c$ where $c = na + nb$ and *nci* is column number of annotation information; $k = nci + c + 2 + j$; N_j is size of library j and $\bar{N}$ is mean of sizes over all libraries; y_{ij} is original count of `RNA` reads in row i and column j.

5 Perform Multiple Beta t-Tests on The Real Data

Suppose the data have been normalized so that all libraries have the same size. After obtaining W value, user can use the function and the real data to perform `mbeta` t-test: `mbetattest`(X,na,nb,W, alpha, file) where X is real data. In our current example, X=*jkttcell*. na and nb are respectively numbers of replicate libraries in conditions 1 and 2. For *jktcell* data, $na = nb = 3$. W is omega value. According to our calculation above step, $W = 1$. *alpha* is



	E	F	G	H	I	J	K	L	M	N	O	P
13395	37126	INTS12	chr4	-	1.1E+08	sg	-1.94354	0.05195	0.80235	0.05641	0.05	-
13396	48100	CCDC84	chr11	+	1.2E+08	tu	1.94994	0.05118	1.26307	0.05572	0.05	-
13397	55001	TMEM49	chr17	+	5.8E+07	sg	-1.95319	0.0508	0.90153	0.05538	0.05	-
13398	58378	KIAA0611	chr20	-	5E+07	sg	-1.95439	0.05065	0.68436	0.05531	0.05	-
13399	34138	PRPF40A	chr2	-	1.5E+08	tu	-1.9725	0.04855	0.73165	0.0531	0.05	+
13400	51790	RAB27A	chr15	-	5.5E+07	tu	-1.98929	0.04667	0.99629	0.05114	0.05	+
13401	55677	PSTPIP2	chr18	-	4.4E+07	sg	2.0056	0.0449	0.79744	0.0493	0.05	+
13402	37611	SAKI	chr5	-	6599356	sg	2.16009	0.03077	0.8255	0.03386	0.05	+
13403	48862	CSRNP2	chr12	-	5.1E+07	tu	2.17827	0.02939	0.81415	0.03243	0.05	+
13404	42480	CPNE3	chr8	+	8.8E+07	tu	-2.24952	0.02448	0.76494	0.0271	0.05	+
13405	51646	ELL3	chr15	-	4.4E+07	sg	-2.26327	0.02362	1.09165	0.02625	0.05	+
13406	39328	HLA-DPA1	chr6	-	3.3E+07	sg	2.3214	0.02027	1.11709	0.02262	0.05	+
13407	31360	ZNF644	chr1	-	9.1E+07	tu	-2.35636	0.01846	0.91156	0.02073	0.05	+
13408	42490	DECR1	chr8	+	9.1E+07	sg	2.41858	0.01558	0.89546	0.01765	0.05	+
13409	59372	LGALS1	chr22	+	3.8E+07	sg	2.81137	0.00493	1.035	0.00566	0.05	+
13410	50230	ATP11A	chr13	+	1.1E+08	tu-ce	-3.15317	0.00162	1.35415	0.0019	0.05	+
13411												

Figure 2: The row number in column A in Figure 1 was deleted. To show explicitly, we here hided `geneid` (for simulation), `tagid` column and selected rows are genes that were identified to be differentially expressed.

the probabilistic threshold. You can set *alpha*=0.05 or 0.01 or the other values; file is *csv* file for saving the results. The example is

```
> res<-mbetattest(X=jkttcell[1:1000,],na=3,nb=3,W=1,alpha=0.05,file="jurkat_NS_48h_tag_mbetattest.csv")
```

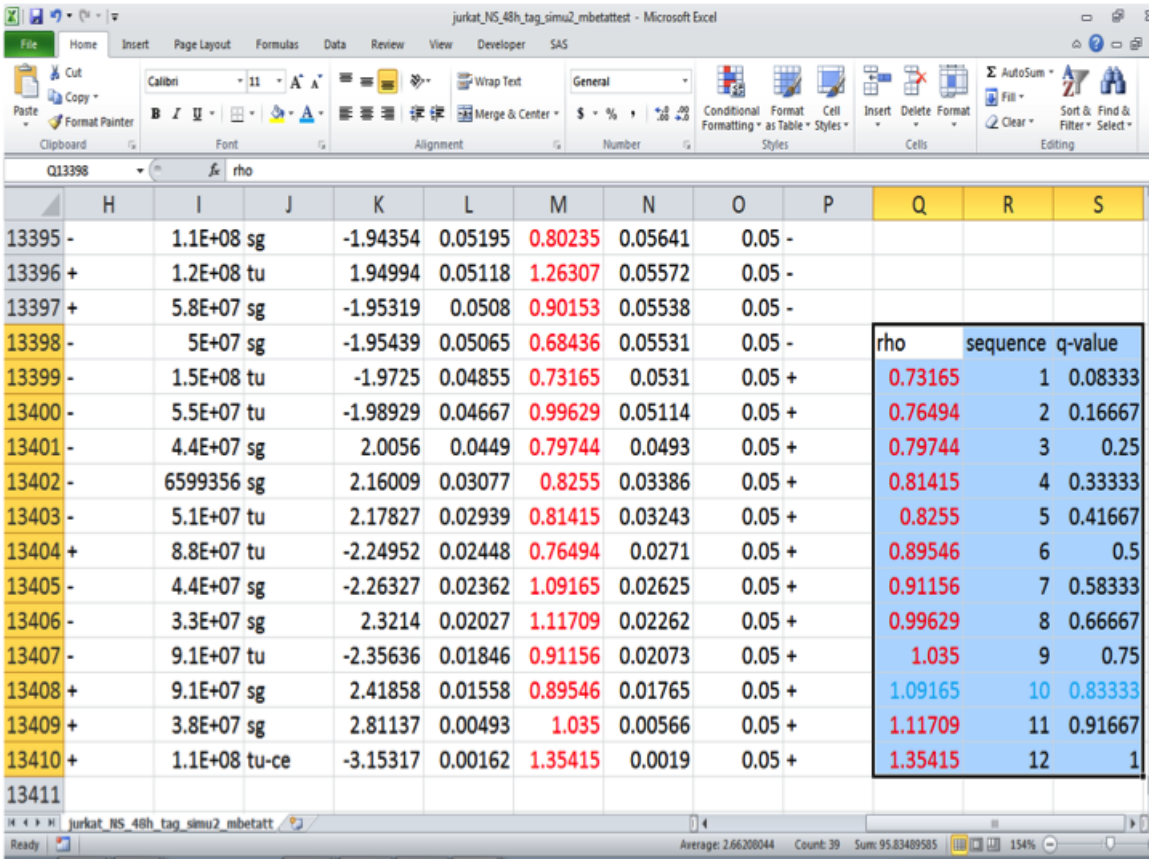
`mbetattest` has two output results: one is saved in *csv* file and the other is dat for *maplot* and for *heatmap*. The package MBttest has this result obtained the whole data. We here load it for making MAplot:

```
> data(dat)
> head(dat)
```

```
   tagid geneid  name   chr strand      pos anno Jurk.NS.A Jurk.NS.B Jurk.NS.C Jurk.48h.A
1  83344  58782   MX1 chr21      + 42831139   sg         0      0.00      0.00      29.61
2 197313  56792  CD22 chr19      + 35838262   sg         0      0.45      3.81      96.33
3 202264  53072  CD19 chr16      + 28950664   sg         0      0.00      0.00      37.65
4 232007  37653 BASP1  chr5      + 17276943   sg         0      0.55      1.31      63.15
5 301820  46661   HBB chr11      -  5246697   sg         0      0.00      0.00       4.52
6 368151  51057 TCL1A chr14      - 96176393   sg         0      1.03      5.15     153.51
  Jurk.48h.B Jurk.48h.C beta_t rho symb
1       0.48          0      0   0    -
2       8.07          0      0   0    -
3       0.52          0      0   0    -
4       0.58          0      0   0    -
5      39.52          0      0   0    -
6       9.93          0      0   0    -
```

```
> maplot(dat=dat,r1=3,r2=3,TT=350,matitle="MA plot")
```

```
> maplot(dat=dat,r1=3,r2=3,TT=50,matitle="MA plot")
```



	H	I	J	K	L	M	N	O	P	Q	R	S
13395	-	1.1E+08	sg	-1.94354	0.05195	0.80235	0.05641	0.05	-			
13396	+	1.2E+08	tu	1.94994	0.05118	1.26307	0.05572	0.05	-			
13397	+	5.8E+07	sg	-1.95319	0.0508	0.90153	0.05538	0.05	-			
13398	-	5E+07	sg	-1.95439	0.05065	0.68436	0.05531	0.05	-	rho	sequence	q-value
13399	-	1.5E+08	tu	-1.9725	0.04855	0.73165	0.0531	0.05	+	0.73165	1	0.08333
13400	-	5.5E+07	tu	-1.98929	0.04667	0.99629	0.05114	0.05	+	0.76494	2	0.16667
13401	-	4.4E+07	sg	2.0056	0.0449	0.79744	0.0493	0.05	+	0.79744	3	0.25
13402	-	6599356	sg	2.16009	0.03077	0.8255	0.03386	0.05	+	0.81415	4	0.33333
13403	-	5.1E+07	tu	2.17827	0.02939	0.81415	0.03243	0.05	+	0.8255	5	0.41667
13404	+	8.8E+07	tu	-2.24952	0.02448	0.76494	0.0271	0.05	+	0.89546	6	0.5
13405	-	4.4E+07	sg	-2.26327	0.02362	1.09165	0.02625	0.05	+	0.91156	7	0.58333
13406	-	3.3E+07	sg	2.3214	0.02027	1.11709	0.02262	0.05	+	0.99629	8	0.66667
13407	-	9.1E+07	tu	-2.35636	0.01846	0.91156	0.02073	0.05	+	1.035	9	0.75
13408	+	9.1E+07	sg	2.41858	0.01558	0.89546	0.01765	0.05	+	1.09165	10	0.83333
13409	+	3.8E+07	sg	2.81137	0.00493	1.035	0.00566	0.05	+	1.11709	11	0.91667
13410	+	1.1E+08	tu-ce	-3.15317	0.00162	1.35415	0.0019	0.05	+	1.35415	12	1
13411												

Figure 3: Demo for calculating W

`myheatmap` has multiple options: both-side, row and column cluster trees with distance methods: "euclidean", "pearson", "spearman", and "kendall" correlation coefficients and color label with "redgreen", "greenred", "redblue", "bluered" or "heat.colors" and angles for genes or isoforms in row and cases (conditions) in column. User can use default without any choice like (Figure **??**) which has tree="both" for both-side tree, or choose tree="column" like (Figure **??**) if columns are species or cancer cases or not choose tree with tree="none" (see (Figure **??**)). User may change *heatmap* color with colors, for example, in (Figure **??**), we chose colors="redblue". If user find that default column name or row name does no have good angles, then user can adjust them with `rwangle` (row angle) or `clangle`(column angle). `rwangle` and `clangle` values are from 0 to 90.

```
> myheatmap(dat=dat,r1=3,r2=3,maptitle="Jurkat T-cell heatmap2")
```

```
> myheatmap(dat=dat,r1=3,r2=3,tree="none",maptitle="Jurkat T-cell heatmap3")
```

```
> myheatmap(dat=dat,r1=3,r2=3,colrs="redblue", tree="column",
+ method="pearson", maptitle="Jurkat T-cell heatmap")
```

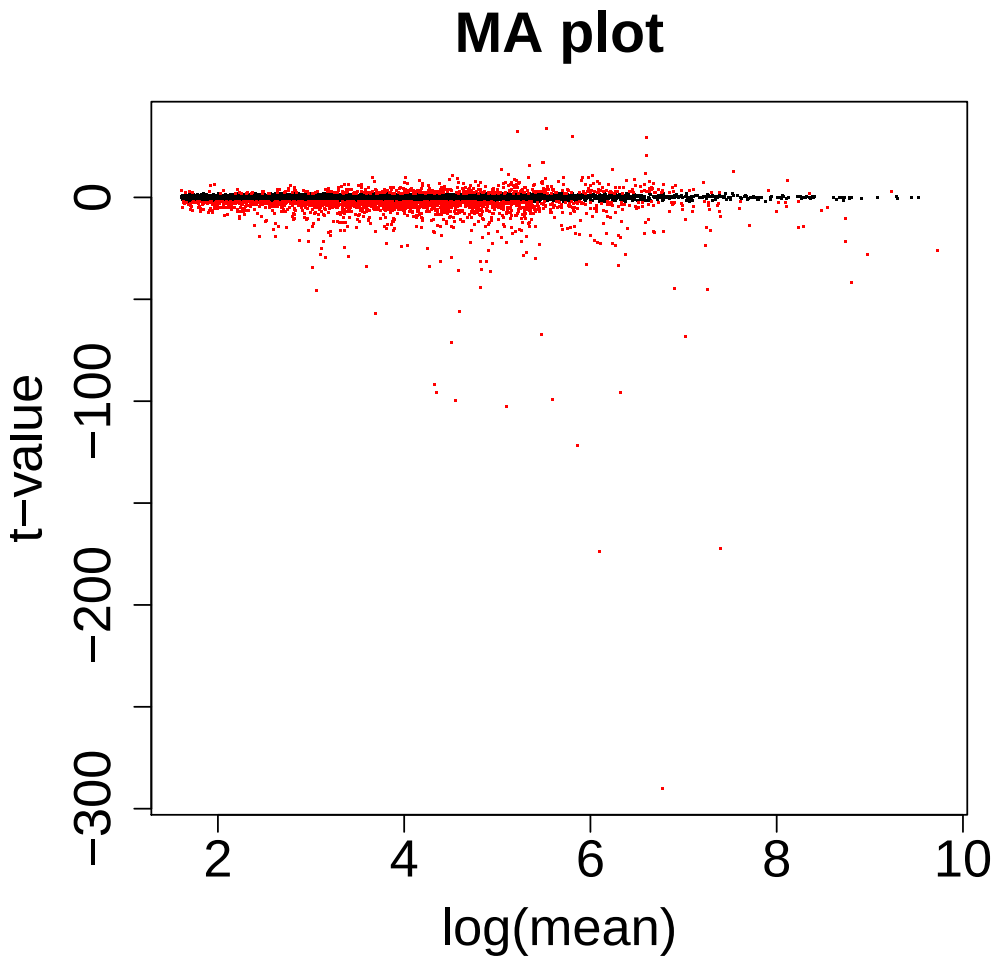


Figure 4: 'MA'-plot of t-value against log mean over all replicate libraries across two conditions. The isoforms who were given differential transcripts in simulation had absolute larger t-values that were highlighted in red than the threshold given in multiple tests. Those who were given no differential expression had very small absolute t-values close to zero labeled in black across long means. Here threshold for truncating t-values is set to be 350, since none of absolute t-values are over 350, the *MAplot* is an outline *MAplot* in which red and black dots are not explicitly seen.

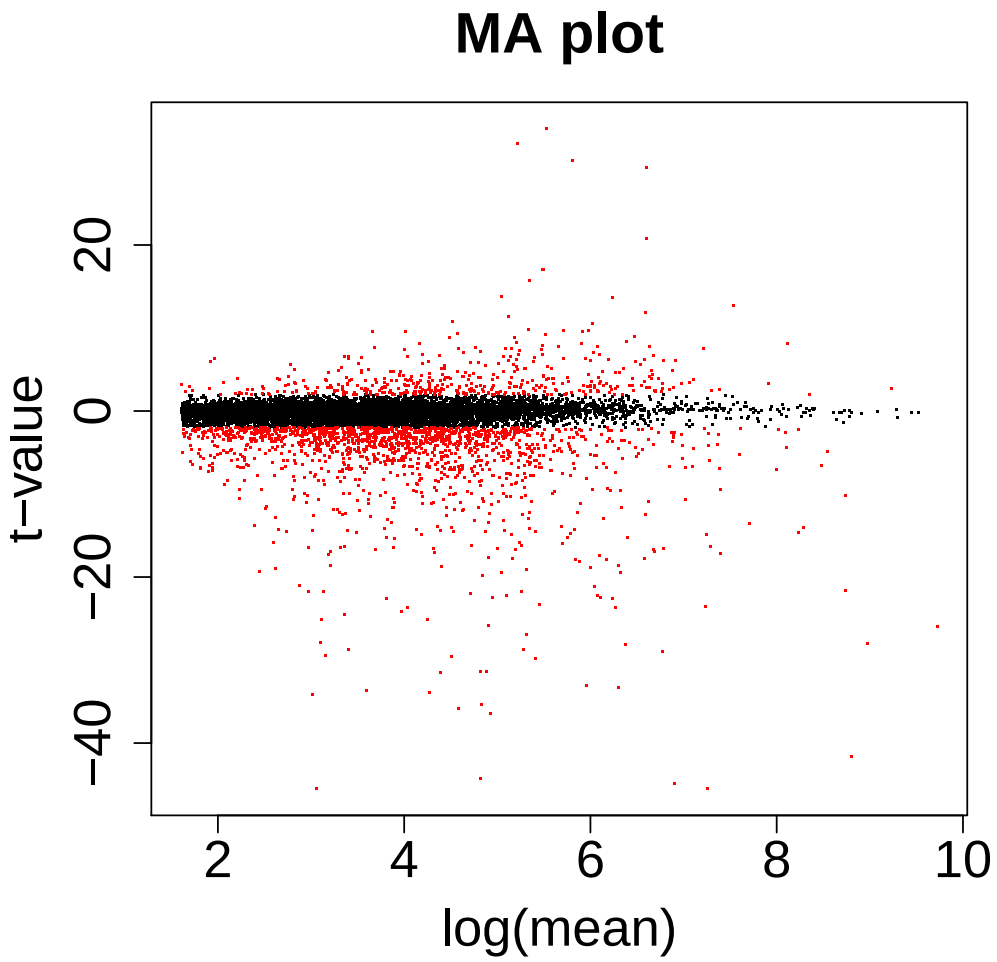


Figure 5: 'MA'-plot of t-value against log mean over all replicate libraries across two conditions. To explicit display the t-values across log mean, absolute t-values $>= 50$ were truncated. One can explicitly see that truly differential transcripts in simulation had absolute larger t-values that were highlighted in red than the threshold given in multiple tests and those who had no differential expressions had very smaller absolute t-values that were labeled in black than the threshold.

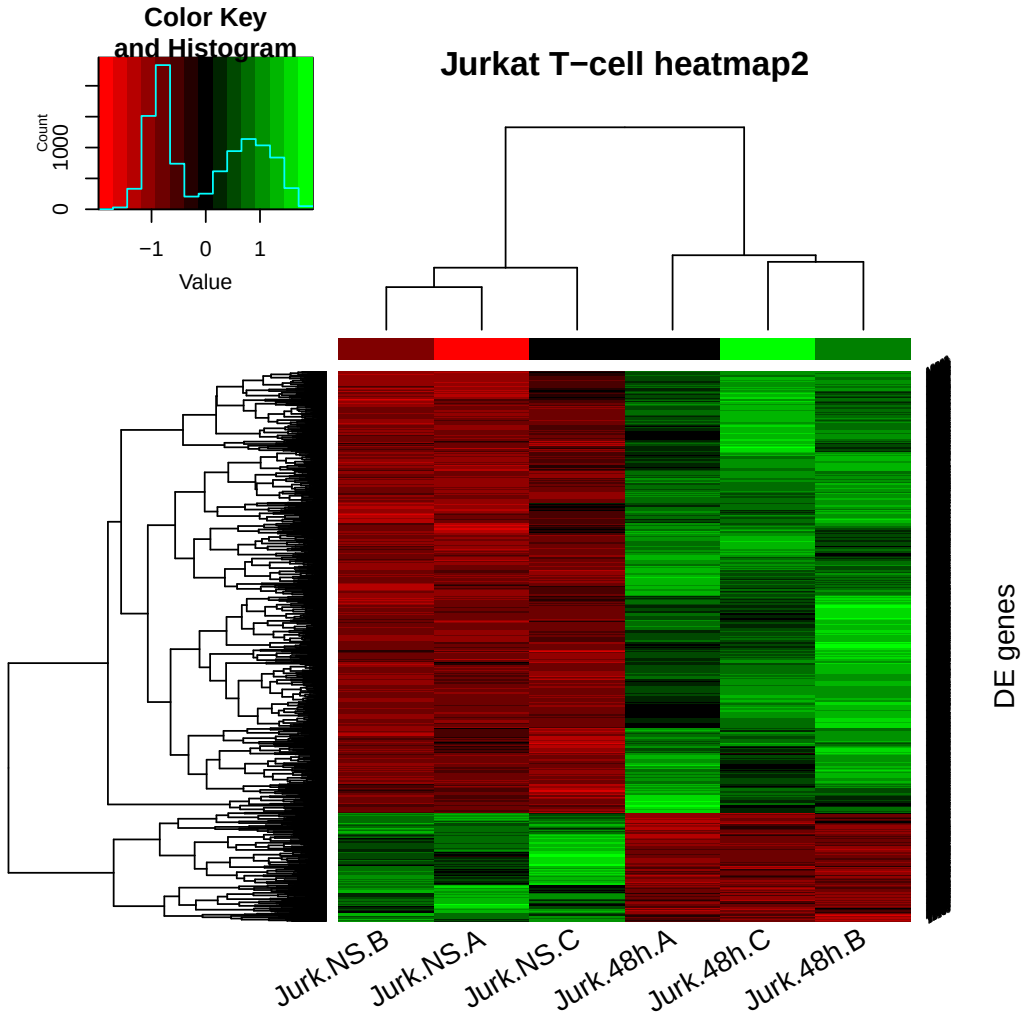


Figure 6: The *heatmap* has both-side trees and displays explicitly differential expression between stimulating and rest. Most of genes were up-expressed by stimulation but a small part of genes were down-expressed. The tree in column divides columns into two groups: `NS` and 48h. The tree in row is tree of differentially expressed genes and also divide genes in row into two big groups.

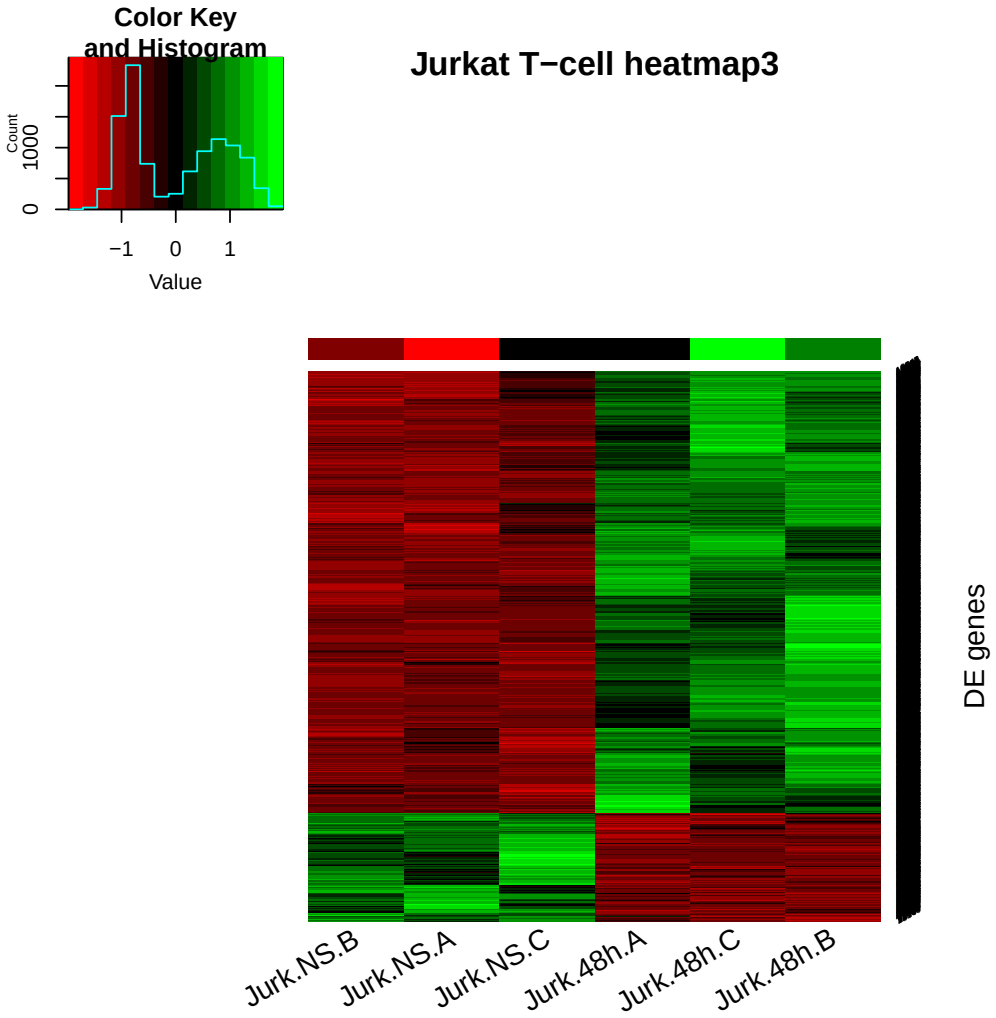


Figure 7: The *heatmap* did not give trees on both sides, but the *heatmap* is the same with (Figure **??**)

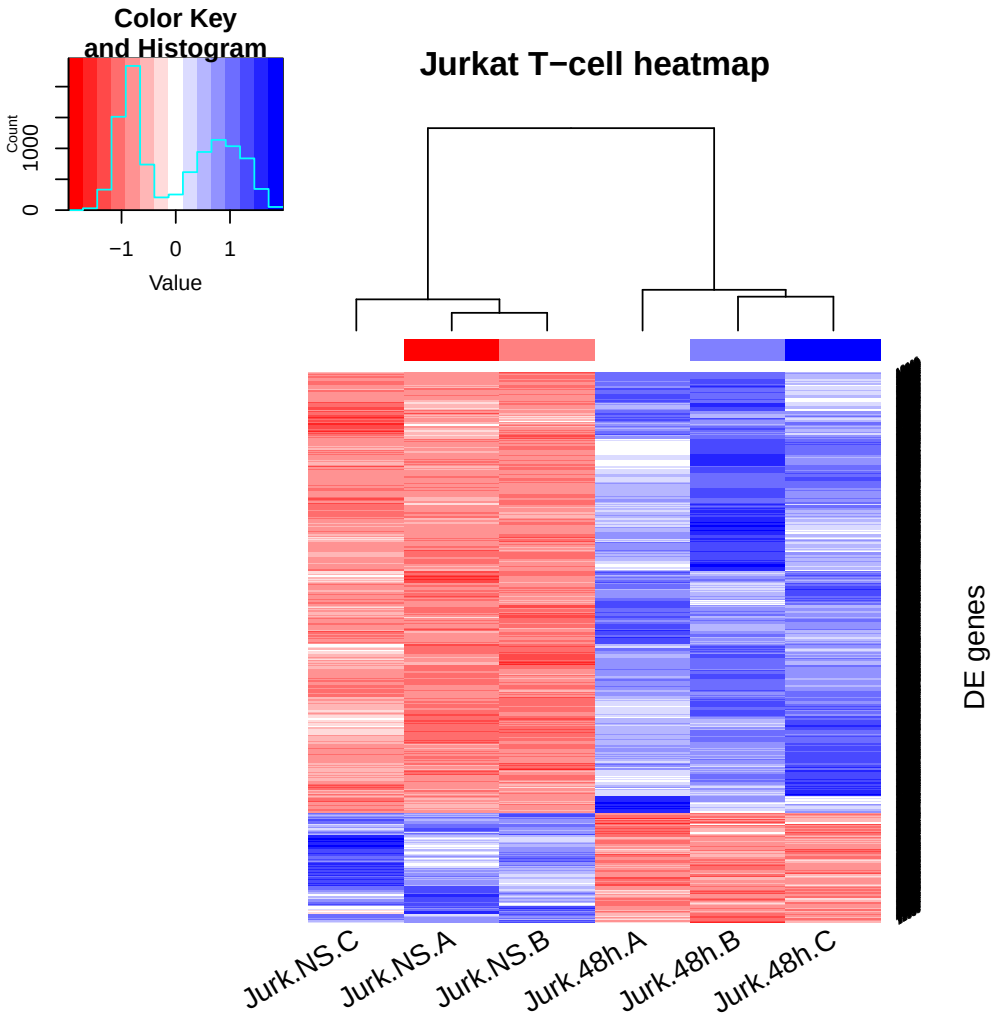


Figure 8: This *heatmap* was labeled with red and blue and gave column trees

6 Session Info

```
> sessionInfo()

R version 3.3.0 RC (2016-04-26 r70550)
Platform: x86_64-apple-darwin13.4.0 (64-bit)
Running under: OS X 10.9.5 (Mavericks)

locale:
[1] C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] MBttest_1.0.0 gtools_3.5.0  gplots_3.0.1

loaded via a namespace (and not attached):
[1] BiocStyle_2.0.0    tools_3.3.0        KernSmooth_2.23-15 gdata_2.17.0
[5] caTools_1.17.1     bitops_1.0-6
```

References

[1] Anders S, Huber W (2010) *Differential expression analysis for sequence count data.* Genome Biol 11: R106.

[2] Thomas J. Hardcastle and Krystyna A. Kelly (2010) *baySeq: Empirical Bayesian Methods For Identifying Differential Expression In Sequence Count Data.* BMC Bioinformatics.

[3] Thomas J. Hardcastle (2015) *Generalised empirical Bayesian methods for discovery of differential data in high-throughput biology.* bioRχv preprint.

[4] Love MI, Huber W, Anders S (2014) *Moderated estimation of fold change and dispersion for RNA-Seq data with DESeq2.* bioRxiv doi:10.1101/002832.

[5] McCarthy DJ, Chen Y, Smyth GK (2012) *Differential expression analysis of multifactor RNA-Seq experiments with respect to biological variation.* Nucleic Acids Res,2012, 40: 4288-4297.

[6] Robinson MD, Smyth GK (2008) *Small-sample estimation of negative binomial dispersion, with applications to SAGE data.* Biostatistics 2008, 9: 321-332.

[7] Robinson MD, McCarthy DJ, Smyth GK (2010) *edgeR: a Bioconductor package for differential expression analysis of digital gene expression data.* Bioinformatics 26: 139-140.

[8] Yuan-De Tan. Anita M. Chandler, Arindam Chaudhury, and Joel R. Neilson(2015) *A Powerful Statistical Approach for Large-scale Differential Transcription Analysis.* Plos One. 2015 DOI: 10.1371.