riborex Manual

Introduction

This is a manual to explain how the riborex is implemented based on current existing methods. The requirement for input data format and parameters used in the implementation are specified in section `Data description`, and the implementation details are described in section `Implementations`. In this manual, we only show a dataset with two conditions, but the pipeline can also be used in data where there are multiple conditions involved.

Packed scripts that can be directly used are available on Github (`link`) for all the implementations mentioned here.

Data description

Raw read counts from both Ribo-seq and RNA-seq are required as input. Here we will use a sample read count table for demonstration. We load the data from file `counts.RData` which is located in the same directory of this manual.

```
load('counts.RData')
RNACntTable <- data$rna
RiboCntTable <- data$ribo
```

We can check the first five lines of the table:

```
head(RNACntTable,5)
```

```
##            total_control_r1 total_control_r2 total_pp242_r1 total_pp242_r2
## uc001upj.2                0                0             0             0
## uc002hhj.3              946             1025           665           753
## uc010pwa.1             1105             1314          1260          1287
## uc001euz.2              464              414           498           458
## uc002jkc.2              216              194           195           228
```

```
head(RiboCntTable,5)
```

```
##            ribo_control_r1 ribo_control_r2 ribo_pp242_r1 ribo_pp242_r2
## uc001upj.2               0               0            0            0
## uc002hhj.3             162            1215          317         1304
## uc010pwa.1             224            1035          306         1416
## uc001euz.2             161             567          161          826
## uc002jkc.2              33             172           36          157
```

Since there are only two conditions in the sample, we specify the numbers of samples for case and control in both Ribo-seq and RNA-seq data. Those numbers will be used for test design in the following section.

```
numCaseRNASmps <- 2
numCtlRNASmps <- 2
numCaseRiboSmps <- 2
numCtlRiboSmps <- 2
```

Implementations

Here we show how riborex is implemented based on edgeR, DESeq2 and voom, given the data and parameters we have in the previous seciton.

Based on edgeR

To begin with, we load the library `edgeR` :

```
library(edgeR)
```

Next, we prepare the factors:

```
condition <- factor(c(rep(0, numCtlRNASmps), rep(1, numCaseRNASmps),
                      rep(0, numCtlRiboSmps), rep(1, numCaseRiboSmps)))
dataType <- factor(c(rep(0, ncol(RNACntTable)), rep(1, ncol(RiboCntTable))))
TE <- factor(c(rep(0, ncol(RNACntTable) + numCtlRiboSmps),
                   rep(1, numCaseRiboSmps))) # Indicator for differential TE
```

Combine Ribo-seq and RNA-seq read count tables and calculate the normalization factors:

```
combCntTbl <- cbind(RNACntTable, RiboCntTable)
dge <- DGEList(counts = combCntTbl)
dge <- calcNormFactors(dge)
```

Then we specify the design matrix and estimate the dispersion for each gene:

```
design <- model.matrix(~condition+dataType+TE)
dge <- estimateDisp(dge, design)
```

The read counts are then fit into GLM, and the likelihood ratio test is performed to detect genes with differential translation efficiency.

```
fit <- glmFit(dge, design)
lrt <- glmLRT(fit)
Results <- topTags(lrt, n=Inf)
```

For example, the 20 most significant genes can be called in this way:

```
topGenes <- Results@.Data[[1]]
head(topGenes,20)
```

```
##                logFC    logCPM       LR       PValue          FDR
## uc002lze.2 -3.487199 10.683804 40.74927 1.730678e-10 3.637539e-06
## uc003phj.2 -3.079697 11.800969 37.27186 1.027569e-09 1.079872e-05
## uc003cfl.3 -2.452908  8.050884 33.41818 7.432595e-09 5.207276e-05
## uc001sfl.2 -2.054399  8.755297 27.88067 1.290321e-07 6.307728e-04
## uc001sbh.3 -2.208747  8.507874 27.55421 1.527525e-07 6.307728e-04
## uc002mjn.2 -2.587955  8.792496 27.17832 1.855278e-07 6.307728e-04
## uc010qrh.1 -4.654739  3.741057 26.93809 2.100775e-07 6.307728e-04
```

```
## uc003axi.2 -2.380761 10.581196 26.59595 2.507685e-07 6.588315e-04
## uc004fkm.2 -2.376734 10.830132 25.94095 3.520220e-07 7.422363e-04
## uc003hyz.2 -2.522914  8.310534 25.93481 3.531432e-07 7.422363e-04
## uc001iou.2 -2.296654 11.305404 24.71747 6.637976e-07 1.166484e-03
## uc001mgs.3 -2.437846  9.666540 24.71111 6.659914e-07 1.166484e-03
## uc003xsm.2 -2.389330  7.990386 24.07487 9.266160e-07 1.498124e-03
## uc004cde.1 -2.228784  9.816837 23.31172 1.377578e-06 1.999056e-03
## uc002vgf.2 -2.305975  9.372143 23.18962 1.467873e-06 1.999056e-03
## uc003bxl.2 -2.260717 10.073795 23.12026 1.521786e-06 1.999056e-03
## uc002qdx.2 -2.187135  9.219604 21.93411 2.821743e-06 3.312786e-03
## uc004ehk.2 -1.914249  8.861306 21.92369 2.837099e-06 3.312786e-03
## uc001ura.2 -2.370001  9.647233 21.71700 3.159784e-06 3.495386e-03
## uc001bhk.2 -2.206106  9.473994 21.49691 3.543996e-06 3.724386e-03
```

The result can be exported:

```
write.table(topGenes[rownames(RNACntTable),], "riborex_edgeR_result.txt", quote=FALSE)
```

Based on DESeq2

First, we load the library `DESeq2`:

```
library(DESeq2)
```

Next, we prepare the factors:

```
condition <- factor(c(rep(0, numCtlRNASmps), rep(1, numCaseRNASmps),
                      rep(0, numCtlRiboSmps), rep(1, numCaseRiboSmps)))
dataType <- factor(c(rep(0, ncol(RNACntTable)), rep(1, ncol(RiboCntTable))))
TE <- factor(c(rep(0, ncol(RNACntTable) + numCtlRiboSmps),
                   rep(1, numCaseRiboSmps)))  # Indicator for differential TE
```

Combine Ribo-seq and RNA-seq read count tables and create design matrix:

```
combCntTbl <- cbind(RNACntTable, RiboCntTable)
combColData <- data.frame(condition = condition)
combColData$dataType <- dataType
combColData$TE <- TE
rownames(combColData) <- colnames(combCntTbl)
dds <- DESeqDataSetFromMatrix(countData = combCntTbl,
                              colData = combColData,
                              design = ~condition + dataType + TE)
```

Finally, we estimate dispersions, fit the data into model.

```
dds <- DESeq(dds)
res <- results(dds)
res
```

```
## log2 fold change (MAP): TE 1 vs 0
## Wald test p-value: TE 1 vs 0
```

```
## DataFrame with 21018 rows and 6 columns
##             baseMean log2FoldChange     lfcSE        stat     pvalue
##            <numeric>      <numeric> <numeric>   <numeric>  <numeric>
## uc001upj.2    0.0000             NA        NA          NA         NA
## uc002hhj.3  675.7641     0.37328696 0.2636464   1.4158622  0.1568159
## uc010pwa.1  825.9618    -0.06879578 0.2530187  -0.2719000  0.7856989
## uc001euz.2  401.5386    -0.20990724 0.2929711  -0.7164776  0.4736965
## uc002jkc.2  124.5855    -0.38437564 0.3064852  -1.2541408  0.2097908
## ...              ...            ...       ...         ...        ...
## uc010yui.1   0.00000             NA        NA          NA         NA
## uc003hru.1 171.93859     0.73424901 0.3000033  2.44746998 0.01438631
## uc002cuk.2  24.99394    -0.16597055 0.3536937 -0.46924930 0.63889145
## uc001dhh.2   0.00000             NA        NA          NA         NA
## uc002lco.2  14.21467    -0.01304869 0.3371263 -0.03870564 0.96912507
##                 padj
##            <numeric>
## uc001upj.2        NA
## uc002hhj.3 0.9959081
## uc010pwa.1 0.9959081
## uc001euz.2 0.9959081
## uc002jkc.2        NA
## ...              ...
## uc010yui.1        NA
## uc003hru.1 0.3350939
## uc002cuk.2        NA
## uc001dhh.2        NA
## uc002lco.2        NA
```

Please note that the rows all set to NAs are those where all counts equal to zero.

The result can be exported:

```
write.table(res[rownames(RNACntTable),], "riborex_DESeq2_result.txt", quote=FALSE)
```

Based on voom

First, we load the library `limma`:

```
library(limma)
```

Next, we prepare the factors:

```
condition <- factor(c(rep(0, numCtlRNASmps), rep(1, numCaseRNASmps),
                      rep(0, numCtlRiboSmps), rep(1, numCaseRiboSmps)))
dataType <- factor(c(rep(0, ncol(RNACntTable)), rep(1, ncol(RiboCntTable))))
TE <- factor(c(rep(0, ncol(RNACntTable) + numCtlRiboSmps),
                    rep(1, numCaseRiboSmps)))  # Indicator for differential TE
```

Combine Ribo-seq and RNA-seq read count tables and calculate the normalization factors:

```
combCntTbl <- cbind(RNACntTable, RiboCntTable)
dge <- DGEList(counts = combCntTbl)
dge <- calcNormFactors(dge)
```

Then we specify the design matrix and apply the voom transformation:

```
design <- model.matrix(~condition+dataType+TE)
v <- voom(dge, design, plot=FALSE)
```

The transformed data is fit into the model and result is obtained:

```
fit <- lmFit(v, design)
fit <- eBayes(fit)
topGenes <- topTable(fit, coef=ncol(design), number=Inf)
```

For example, the 20 most significant genes can be called in this way:

```
head(topGenes,20)
```

```
##                logFC   AveExpr          t      P.Value    adj.P.Val
## uc002lze.2 -3.599639 10.194260 -10.756150 5.772673e-27 1.213300e-22
## uc003phj.2 -3.180055 11.428293  -9.577713 1.017851e-21 1.069660e-17
## uc003qdx.2 -3.344340  8.729456  -9.425054 4.403146e-21 3.084844e-17
## uc002pny.2 -3.290670  8.613579  -9.322057 1.167557e-20 6.134926e-17
## uc004bqy.1 -3.261109  7.691228  -8.674196 4.237384e-18 1.781227e-14
## uc001fdv.2 -2.896751  8.680406  -8.285946 1.188751e-16 4.164195e-13
## uc002nhp.1 -2.698963  9.488641  -7.969285 1.615798e-15 4.851549e-12
## uc002mjn.2 -2.744068  8.407552  -7.847258 4.302089e-15 1.130266e-11
## uc003hyz.2 -2.736047  7.947694  -7.651608 2.005830e-14 4.684282e-11
## uc010ygb.1 -2.561545  8.836247  -7.559649 4.082384e-14 8.580354e-11
## uc001mgs.3 -2.554519  9.246664  -7.511240 5.914638e-14 1.130126e-10
## uc002cno.2 -2.869726  7.645022  -7.331087 2.303355e-13 4.034327e-10
## uc004fkm.2 -2.415989 10.531870  -7.286196 3.216130e-13 5.199740e-10
## uc001ura.2 -2.440391  9.294297  -7.222272 5.155701e-13 7.740181e-10
## uc003axi.2 -2.407556 10.145362  -7.205423 5.834715e-13 8.175602e-10
## uc001iou.2 -2.330113 11.152153  -7.064330 1.626394e-12 2.136471e-09
## uc003yyt.2 -2.368427  9.331394  -7.050803 1.792524e-12 2.216193e-09
## uc002vgf.2 -2.381683  9.035003  -7.001557 2.550209e-12 2.680015e-09
## uc003xsm.2 -2.504466  7.703804  -7.006567 2.460636e-12 2.680015e-09
## uc003cfl.3 -2.522534  7.729335  -7.006548 2.460972e-12 2.680015e-09
##                    B
## uc002lze.2 49.66247
## uc003phj.2 38.08469
## uc003qdx.2 36.55051
## uc002pny.2 35.63080
## uc004bqy.1 29.89716
## uc001fdv.2 26.85017
## uc002nhp.1 24.40322
## uc002mjn.2 23.43598
## uc003hyz.2 21.94941
## uc010ygb.1 21.32768
## uc001mgs.3 20.97188
## uc002cno.2 19.53577
## uc004fkm.2 19.38048
## uc001ura.2 18.91891
## uc003axi.2 18.80893
## uc001iou.2 17.84220
```

```
## uc003yyt.2 17.73924
## uc002vgf.2 17.39698
## uc003xsm.2 17.39630
## uc003cfl.3 17.39049
```

The result can be exported:

```
write.table(topGenes[rownames(RNACntTable),], "riborex_voom_result.txt", quote=FALSE)
```