DEG Analysis, including DESeq, DESeq2, edgeR, and limma

Calvin WC Chan, Ph.D.[*1]

[1]Center for Research Informatics, University of Chicago, Chicago, IL 60637

*wchan10@bsd.uchicago.edu

29 August 2018

Abstract

- Summary
  - x
- Tutorial
  - DESeq
  - DESeq2
  - edgeR
  - limma

Package

DEG Analysis, including DESeq, DESeq2, edgeR, and limma

Contents

1 Summary 2

2 Tutorial 2

2.1 Dataset - airway 2

2.2 DESeq Pipeline R Documentation 3

2.3 DESeq2 Pipeline Bioconductor 4

2.4 edgeR Pipeline R Documentation 5

2.5 limma Pipeline R Documentation 6

```
# Not Run
'''{}'''.format(
  "this is a python snippet"
)
```

1 Summary

- The use of **design matrix** & **contrast**; for explicit expression string (e.g., GroupKO or GroupWT), now using the design matrix as ~ 0 + Group
  - `~ 0 + Group & edgeR::glmLRT(contrast = grepl(c.case.curated, colnames(DGEGLM.fit$design))*1 - grepl(c.ctrl.curated, colnames(DGEGLM.fit$design))*1)`
  - `~ 0 + Group & DESeq2::results(contrast = as.list(paste0(c(formula = "Group"), c(numerator = c.case.curated, denominator = c.ctrl.curated))))`
  - `~ 0 + Group & limma::contrasts.fit(fit = MArrayLM.c, contrasts = limma::makeContrasts(contrasts = paste(c.case.curated, c.ctrl.curated, sep = "-"), levels = mat.design))`
    - here, REMOVE MAD (Median Absolute Deviation) as zero in MArrayLM.t to generate MArrayLM.c
- Result
  - edgeR::topTags(): Geneid, logFC, logCPM, F, PValue, FDR
  - DESeq2::results(): baseMean, log2FoldChange, lfcSE, stat, pvalue, padj
  - limma::topTable(): Geneid, logFC, AveExpr, t, P.Value, adj.P.Val, B

2 Tutorial

2.1 Dataset - airway

```
data(airway)
se <- airway
head(assay(se))
##                 SRR1039508 SRR1039509 SRR1039512 SRR1039513 SRR1039516
## ENSG00000000003        679        448        873        408       1138
## ENSG00000000005          0          0          0          0          0
## ENSG00000000419        467        515        621        365        587
## ENSG00000000457        260        211        263        164        245
## ENSG00000000460         60         55         40         35         78
## ENSG00000000938          0          0          2          0          1
##                 SRR1039517 SRR1039520 SRR1039521
## ENSG00000000003       1047        770        572
## ENSG00000000005          0          0          0
## ENSG00000000419        799        417        508
## ENSG00000000457        331        233        229
```

```
## ENSG00000000460         63         76         60
## ENSG00000000938          0          0          0
```

```
y <- edgeR::DGEList(
  counts = round(assay(se))
)

sum(
  keep <- apply(
    edgeR::getCounts(y),
    1,
    function(x) { all(x >= 10) }
  )
)
## [1] 14221

y <- y[keep, , keep.lib.size = FALSE]
```

2.2 DESeq Pipeline R Documentation

- The count table + The metadata (to CountDataSet$condition)
  - `CountDataSet <- DESeq::newCountDataSetFromHTSeqCount(sampleTable, directory)`
- (additional) Filtering for gene type and/or lowly expressed genes
  - on-hand script
- Normalziation
  - `CountDataSet.sizeFact <- DESeq::estimateSizeFactors(CountDataSet)`
  - (additional) Saving normalized count table
    - DESeq::counts(CountDataSet.sizeFact.estiDisp, normalized = TRUE)
- Variance estimation
  - Working with replicates or partially without replicates
    - CountDataSet.sizeFact.estiDisp <- DESeq::estimateDispersions(CountDataSet.sizeFact)
  - Working without any replicates
    - CountDataSet.sizeFact.estiDisp <- DESeq::estimateDispersions(CountDataSet.sizeFact, method = "blind", sharingMode = "fit-only")
  - (exploratory) Ploting dispersion estimates and fitted values
    - DESeq::plotDispEsts(CountDataSet.sizeFact.estiDisp)
- Inference: Calling differential expression
  - CountDataSet.sizeFact.estiDisp.NBTest <- DESeq::nbinomTest(CountDataSet.sizeFact.estiDisp, ctrl, case)
    - This function tests for differences between the base means of **two conditions** (i.e., for differential expression in the case of RNA-Seq).
    - (additional) Ploting M (log ratio) and A (mean average) plot
      - DESeq2::plotMA(df[, ("baseMean", "log2FoldChange"], pari, de.tags)
    - (additional) Ploting a histogram of unadjusted p-values of all genes
    - (additional) Ploting a histogram of adjusted p-values (q-value / FDR) of all genes

2.3 DESeq2 Pipeline Bioconductor

- The count table + The metadata (to DESeqDataSet$condition)
  - DESeqDataSet <- DESeq2::DESeqDataSetFromMatrix(sampleTable, directory, design), or
  - DESeqDataSet <- DESeq2::DESeqDataSetFromHTSeqCount(sampleTable, directory, design)
- (additional) Filtering for gene type and/or lowly expressed genes
  - on-hand script
  - DESeq2::fpm(DESeqDataSet, robust = FALSE)
  - DESeq2::counts(DESeqDataSet) ~ DESeqDataSet@assays[["counts"]]
    - DESeqDataSet@assays ~ SummarizedExperiment::assay(DESeqDataSet)
- Normalziation + Variance estimation + Inference: Calling differential expression
  - LOG
    - estimateSizeFactors (estimating size factors),
    - estimateDispersions (estimating dispersions, gene-wise dispersion estimates, mean-dispersion relationship, final dispersion estimates),
    - and then nbinomWaldTest (fitting model and testing)
  - DESeqDS.sizeFact.estiDisp.NBTest <- DESeq2::DESeq(DESeqDataSet)
    - @ fitType = c("parametric", "local", "mean"), # either "parametric", "local", or "mean" for the type of fitting of dispersions to the mean intensity
      - For decreasing gene-wise dispersion estimates over mean (using plotDispEsts) one should use parametric, unless the parametric fitting procedure does not work, in which case use "local" (local regression is actually automatically substituted with a message in the case that the parametric fitting procedure does not converge.) The "mean" option is useful when there is no apparent dependence of dispersion estimates over mean (using plotDispEsts). This choice does not depend on sample size, but on the apparent dependence of the gene-wise estimates (the MLE for each gene) on the mean of counts.
      - If you are referring to the the tagwise estimation in edgeR, the tagwise estimation is similar to the estimateDispersionsMAP() step in DESeq2, which is the last step in estimateDispersions(), which is automatically used by DESeq()
    - paste(colnames(attr(DESeqDS.sizeFact.estiDisp.NBTest, "dispModelMatrix")), collapse = ",") ~ resultsNames(DESeqDS.sizeFact.estiDisp.NBTest)
    - (additional) Saving normalized count table
      - DESeq2::counts(DESeqDS.sizeFact.estiDisp.NBTest, normalized = TRUE)
    - (additional) Saving **transformed**, normalized count table
      - `SummarizedExperiment::assay(DESeqTransform)` return a matrix
      - `DESeqTransform.ntd <- DESeq2::normTransform(DESeqDS.sizeFact.estiDisp.NBTest, f = log2, pc = 1)`
      - `DESeqTransform.vst <- DESeq2::vst(DESeqDS.sizeFact.estiDisp.NBTest, blind = FALSE, nsub = min(1000, nrow(DESeqDS.sizeFact.estiDisp.NBTest)))`
      - `DESeqTransform.rld <- DESeq2::rlog(DESeqDS.sizeFact.estiDisp.NBTest, blind = FALSE)`
    - (exploratory) Ploting the per-gene dispersion estimates together with the fitted mean-dispersion relationship
      - DESeq2::plotDispEsts(DESeqDS.sizeFact.estiDisp.NBTest)
- Extracting results from a DESeq2 analysis

- DESeqRes <- DESeq2::results(CountDataSet.sizeFact.estiDisp, contrast)
  - (additional) Ploting M (log ratio) and A (mean average) plot
    - DESeq::plotMA(df[, ("baseMean", "log2FoldChange"], pair, de.tags)
    - Since DESeq::plotMA() doesn't have the col argument as DESeq::plotMA(), in which we can define DEGs by our own
  - (additional) Ploting a histogram of unadjusted p-values of all genes
  - (additional) Ploting a histogram of adjusted p-values (q-value / FDR) of all genes

2.4 edgeR Pipeline R Documentation

- The count table + The metadata (to DGEList$samples)
  - DGEList <- edgeR::DGEList(counts, samples, genes, remove.zeros), or
  - DGEList <- edgeR::readDGE(files, path, columns=c(1,2), group, labels[, header, sep])
- (additional) Filtering for gene type and/or lowly expressed genes
  - on-hand script
  - edgeR::cpm(DGEList, normalized.lib.sizes = FALSE)
  - (**just raw counts without taking normalized.lib.sizes into account!!!**) edgeR::getCounts(DGEList) ~ DGEList$counts
  - get normalized counts by `t(t(DGEList.sizeFact$counts) * DGEList.sizeFact$samples$norm.factors)`
- Normalziation
  - DGEList.sizeFact <- edgeR::calcNormFactors(DGEList, method = "TMM")
  - (additional) Saving normalized count table
    - t(t(DGEList.sizeFact$counts) x DGEList.sizeFact$samples$norm.factors)
- Variance estimation
  - Working with replicates or partially without replicates
    - DGEList.sizeFact.estiDsip <- edgeR::estimateDisp(DGEList.sizeFact, design)
  - Working without any replicates
    - CONSIDER bulk RNA-seq data as a whole to estimate dispersions
  - (exploratory) Ploting dispersion estimates and fitted values
    - edgeR::plotBCV(DGEList.sizeFact.estiDsip)
- Inference: Calling differential expression
  - Working with replicates or partially without replicates
    - DGEGLM.fit <- edgeR::glmQLFit(DGEList.sizeFact.estiDsip)
      - glmQLFit and glmQLFTest implement the quasi-likelihood (QL) methods of Lund et al (2012), with some enhancements and with slightly different glm, trend and FDR methods. See Lun et al (2015) for a tutorial describing the use of glmQLFit and glmQLFit as part of a complete analysis pipeline.
      - glmQLFTest is similar to glmLRT except that it replaces likelihood ratio tests with empirical Bayes quasi-likelihood F-tests. The p-values from glmQLFTest are always greater than or equal to those that would be obtained from glmLRT using the same negative binomial dispersions.
      - CONCLUSINO: edgeR implements the Negative Binomial Generalized Linear Model (glm) with two likelihood ratio test, which are glmQLFit/glmQLFTest and glmFit/glmLRT. However, glmQLFit/glmQLFTest is stringent then glmFit/glmLRT.
  - Working without any replicates

        - CONSIDER bulk RNA-seq data as a whole to estimate dispersions will cause an ERROR when calling edgeR::glmQLFit() -Error in squeezeVar(s2, df = df.residual, covariate = AveLogCPM, robust = robust, : Could not estimate prior df
    - DGELRT.res <- edgeR::glmQLFTest(DGEGLM.fit, contrast)
        - (additional) Ploting M (log ratio) and A (mean average) plotedgeR::plotSmear
            - DESeq::plotMA(df[, ("baseMean" <- 2^logCPM, "log2FoldChange" <- "logFC"], pair, de.tags)
        - (additional) Ploting a mean-difference plot of two libraries of count data with smearing of points with very low counts, especially those that are zero for one of the columns.
            - edgeR::plotSmear(, pair, de.tags)
        - (additional) Ploting a histogram of unadjusted p-values of all genes
        - (additional) Ploting a histogram of adjusted p-values (q-value / FDR) of all genes
- Extracting results from a edgeR analysis
    - TopTags <- edgeR::topTags(DGELRT.res, n = Inf)
        - **!!! NEED to remove those genes with NaN values in fold change !!!**

2.5 limma Pipeline R Documentation

- The count table + The metadata (to DESeqDataSet$condition)
    - DGEList <- edgeR::DGEList(counts, samples, genes, remove.zeros), or
    - DGEList <- edgeR::readDGE(files, path, columns=c(1,2), group, labels[, header, sep])
- (additional) Filtering for gene type and/or lowly expressed genes
    - on-hand script
    - edgeR::cpm(DGEList, normalized.lib.sizes = FALSE)
    - edgeR::getCounts(DGEList) ~ DGEList$counts
- Normalziation
    - DGEList.sizeFact <- edgeR::calcNormFactors(DGEList, method = "TMM")
    - (additional) Saving normalized count table
        - t(t(DGEList.sizeFact$counts) x DGEList.sizeFact$samples$norm.factors)
- Variance estimation
    - Transform RNA-Seq Data Ready for Linear Modelling
        - `EList.voom <- limma::voom(counts = DGEList.sizeFact, design, plot = TRUE, save.plot = TRUE, normalize.method = "quantile", span = 0.5)`
        - (additional) Saving **transformed**, normalized count table
- Inference: Calling differential expression
    - Working with replicates
    -
    - MArrayLM <- limma.lmFit(object, design, method = "ls")
    - **if multiple comparisons**
        - `mat.contrast <- limma::makeContrasts(contrasts = contrasts = paste(c.case.curated, c.ctrl.curated, sep = "-"), levels = mat.design)`
        - MArrayLM <- **limma::contrasts.fit**(fit = MArrayLM, contrasts = mat.contrast)
    - MArrayLM <- limma::eBayes(fit = MArrayLM, proportion = 0.01, stdev.coef.lim = c(0.1, 4), trend = F, robust = F, winsor.tail.p = c(0.05, 0.1))

        - (additional) Ploting the quantiles of a data sample against the theoretical quantiles of a Student's t distribution
            - limma::qqt()
        - (additional) Multiple Testing Across Genes and Contrasts
            - limma::decideTests()
        - (additional) Ploting M (log ratio) and A (mean average) plot
            - DESeq::plotMA(df[, ("baseMean", "log2FoldChange"], pair, de.tags)
            - Since DESeq::plotMA() doesn't have the col argument as DESeq::plotMA(), in which we can define DEGs by our own
    -
    - Working partially without replicates or without any replicates
        - **Now, no solution for using limma on a data set without replicate**
- Extracting results from a limma analysis
    - limma::topTable()
        - (additional) Ploting a histogram of unadjusted p-values of all genes
        - (additional) Ploting a histogram of adjusted p-values (q-value / FDR) of all genes