RNA-seq Differential Expression (DE) Analysis Using DESeq2

Jessica Randall

Briefly, DESeq2 uses a Wald test to determine differential gene expression between at least two groups. DESeq2 assumes a negative binomial model which mathematically accounts for the fact that we are assessing gene counts and we are assuming that most genes we are comparing between the groups will not be differential expressed.

Linked is the original paper from Anders & Huber introducing the concepts implemented in DESeq2. We will be using the pasilla package for our example data.

Please note that DESeq2 has a number of capabilities that we will not be covering. You can import data for use in DESeq2 in several different ways, there are options for single cell projects, for incorporating Bayesian statistics, for time-series experiments, for outliers, for obtaining all of the results that DESeq2 functions produce, and even more options for graphing (including an R shiny app we plan to cover in a future walk-through) so we strongly encourage you to reach out to EICC with questions regarding options available to you with DESeq2. Check out some of the graphs from previous projects here.

Definition of terms

0.0.0.1 Wald test: DESeq2 offers two options with how it determines differential expression, the Wald test and the Likelihood Ratio Test. Users of edgeR may be familiar with the latter. In DESeq2, the default test for pairwise comparison analysis is called the Wald Test and it is looking at your control and your experimental samples to see if the difference between them is equal to zero or not.

Compared to edgeR it has been our experience that DESeq2 is more liberal in its calling of genes as significantly DE. If you are not sure which or how many DE genes you are expecting to find in your experiment, DESeq2 would likely the best analysis tool for your data.

0.0.0.2 Unadjusted p values vs adjusted p-values/(FDR): In DE analysis, a single p-value tells you how likely it is that a single gene is differentially expressed between at least two groups (ex: a control and a treatment group) due to some actual difference between the groups as opposed to random chance. False Discovery Rate (FDR) tells you how likely it is that all genes identified as DE are false positives. A FDR of 5% means that among all genes called DE, an average of 5% of those are truly not DE. DE genes are only considered significantly so if they meet the adjusted p value, not only the unadjusted p-value. FDRs for each individual gene are called q-values or local FDRs.

Loading data

Our very first step is to load the libraries we'll need to assess the functions required for analysis and graphing. Please see Bioconductor for information about initial installation and use of Bioconductor and its packages. We also set the minimal theme in gglot2 for all graphs to have the same aesthetic features by default.

The pasilla experiment studied RNAi knockdown of Pasilla, the Drosophila melanogaster ortholog of mammalian NOVA1 and NOVA2, on the transcriptome. Data are provided by NCBI Gene Expression Omnibus under accession numbers GSM461176 to GSM461181.

DESeq2 offers many options for importing count data and data about your samples. Here we will demonstrate importing the count matrix and sample data from the pasilla package since we're using it as an example. Typically we will use the here package to specify the path for the counts and sample data files in a list of files to import and export from the task.

We're also going to specify that we'd like the row names of our sample data to come from the first column, called "file" since this is where we've stored which sample is which and finally we remove extra columns from our sample data which we won't be using in our analysis.

Please reach out to EICC if you would like to compare 3 or more groups as this is a simplified example. It may also be the case you will need more than 6 samples per experimental group or that you may need to remove genes with average counts greater than 5, 10, 15, or even 20 for sufficient statistical power. Please see our PROPER walk-through for an example of our of power and sample size analysis.

Our data is almost ready to analyze but first, DESeq2 requires that the row names of the sample data are the same as the column names of the count data. This is why we used the informal unit test below to check before proceeding with analysis. In order to do the matrix multiplication as part of the analysis, we need to have the samples in the counts file be labelled in the same way they are in the first column of the sample data file, typically this is where sample ids or abbreviated sample names are stored.

```
rownames(sampledata) <- sub("fb", "", rownames(sampledata))

countdata <- countdata[,rownames(sampledata)]

stopifnot(rownames(sampledata) %in% colnames(countdata))
```

Now that we know this is true, we can proceed.

Preparing for Analysis

In order to preform a differentially expression analysis, we need to specify some information about our data. In DESeq2 we must create a special object called a DESeqdataset object, here abbreviated as dds. This object takes in the count data, the sample data, and the variable we would like to compare between the samples as inputs.

Next, we specify that the untreated group is our reference group to which we would like to compare our treated samples.

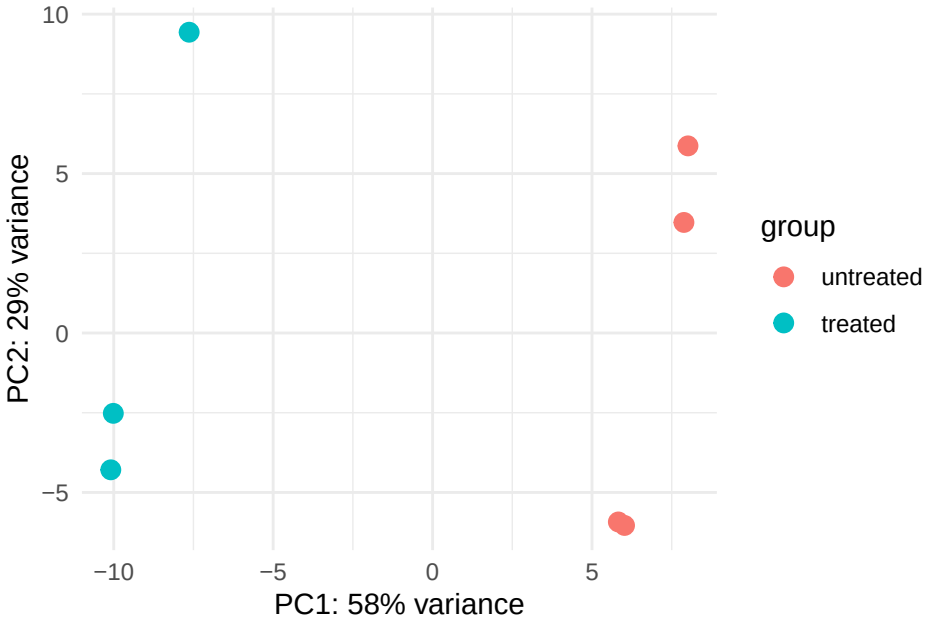
```
dds <- DESeqDataSetFromMatrix(countData = countdata,
                              colData = sampledata,
                              design = ~ condition)

dds$condition <- relevel(dds$condition, ref = "untreated")
```

If we had any additional data to add about the samples that we wanted to include in our analysis we would add it next but since this is a simplified example, we are only comparing treated and control samples without taking into account any additional information about them.

At this point we generate our first exploratory visualization, the principal components analysis plot. This will show us how your data cluster or how similar each sample is to others of the same group. There are percentages along the axes and the percentage on the x-axis tells us how much the differences between the samples is explained by them being treated or untreated.

We start by transforming our data using the variance stabilizing transformation available from the vsn library (Tibshirani 1988; Huber et al. 2003; Anders and Huber 2010). This is similar to using a log2 transformation with normally distributed data with many very small values. VST adjust the data such that if the means of the rows are small, as they often are in gene counts, the variance will remain relatively constant across all all counts. Doing this allows the user to cluster the samples into experimentally interesting groups in graphs rather than seeing groups clustered by their variance. We then typically save this as a data frame to export to clients.

We can use the same plotPCA function to obtain the coordinates for each sample on the plot. This is helpful in identifying samples we would consider outliers since we haven't labelled each sample on the graph.



We would interpret this as the samples being somewhat clustered clearly by group and interpret the percentage on the x-axis as 58% of the variability between these samples is due to them being treated or untreated. I might also say that the within-group variability between samples in the treated group is probably contributing some noise to our ability to detect differences between the treated and untreated groups. The y-axis tells us how much variability between these samples is due to other factors in our model or if we have none, sources of variability we may not have accounted for like sex or ethnicity which are often leading contributors of variability between samples and should be accounted for in experimental design if you wish to control for their effects.

Prefiltering

Typically we want to ignore genes that have counts of zero across all samples since these are adding statistical noise. We may also want to be more stringent and remove genes with rows that sum to 10, 20 or even 30 or less since these could also be contributing noise.

DESeq2 will filter genes it deems as low counts automatically based on the sum of the mean-normalized counts in each row. We'll see the criteria chosen when we view the results of our analysis later on. The results will tell you how many genes were removed and how many remain. If you would like to specify your own cut-offs for filtering or if you do not want DESeq2 to do any additional filtering, these are parameters that can be adjusted.

If you choose to do DE analysis through EICC we typically rely on DESeq2's robust filtering since it tends to increase power to detect DE genes but we would customize this part of the analysis based on your data should you choose to do so.

Testing

DESeq2's analysis step is almost deceptively user-friendly compared to the analysis steps of edgeR and baySeq. It will tell you the steps it is taking with your data and you have the option to ask for additional output and customization but it keeps necessary user input to a minimum for most simple experiments.

The default analysis explained here is the use of the Wald test. DESeq2 also offers the option of the Likelihood Ratio test. Both of these tests rely on the assumption that your count data follow a negative binomial distribution which means that we assume that most counts are very low and that there are more non-DE genes between the groups than there are DE genes. Which test we choose will depend on your experimental design and properties of your data. Our summary presentations for clients typically include information on the model we chose, justification, and the null and alternative hypotheses of that model.

Since we want to be especially sure we are comparing our treated group to the untreated group (and not the over way around) we use the resultsNames function (not shown) to identify the comparisons available and select the one we'd like to see. In this case, based on condition, we want to see the results of the treated vs untreated. If we had other groups, we would see other options in the place of the "treated" group but since we set our reference group to "untreated" all comparisons would have that group listed second. This avoids having to re-run the DESeq function at every new comparison desired.

Next, we create our results with the results function, use the summary function to see a tabular summary of them, and save them as a data frame for further manipulation. In the results function we also specify that we would like to set the FDR to 0.05. By setting the FDR here, we can experiment with different cut-offs based on what we're willing to accept.

The summary gives you information about the total number of genes with non-zero read counts, the FDR specified above, the exact number and total percentage of the up and down regulated DE genes, the presence of any outliers, and the removal of any additional genes with low counts.

```
##
## out of 12359 with nonzero total read count
## adjusted p-value < 0.05
## LFC > 0 (up)       : 406, 3.3%
## LFC < 0 (down)     : 432, 3.5%
## outliers [1]       : 1, 0.0081%
## low counts [2]     : 3797, 31%
## (mean count < 5)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

Results

In this example we see that while controlling the adjusted p/FDR threshold at <0.05, we have 838 DE genes between these groups, 406 are more expressed in the treated samples or up-regulated and 432 are more expressed in the untreated samples or down-regulated. Up and down regulation refers to the group you have set as the control or reference group. In this case, we are comparing the treated samples to the untreated samples so the untreated samples are our reference group and we say genes are up or down regulated in comparison to this group.

In our walkthorugh of edgeR we use the same example dataset and found that while controlling the adjusted p/FDR threshold at <0.05, we found no differentially expressed genes between these groups.

Different analytical tools will often give you slightly different results and edgeR may be better for projects which have specified genes of interest in mind rather than exploratory projects since edgeR is much more strict with potential false positive results. While DESeq2 may allow more false positives into your significant results it also provides additional tools for evaluating their veracity before following up with a lab test.

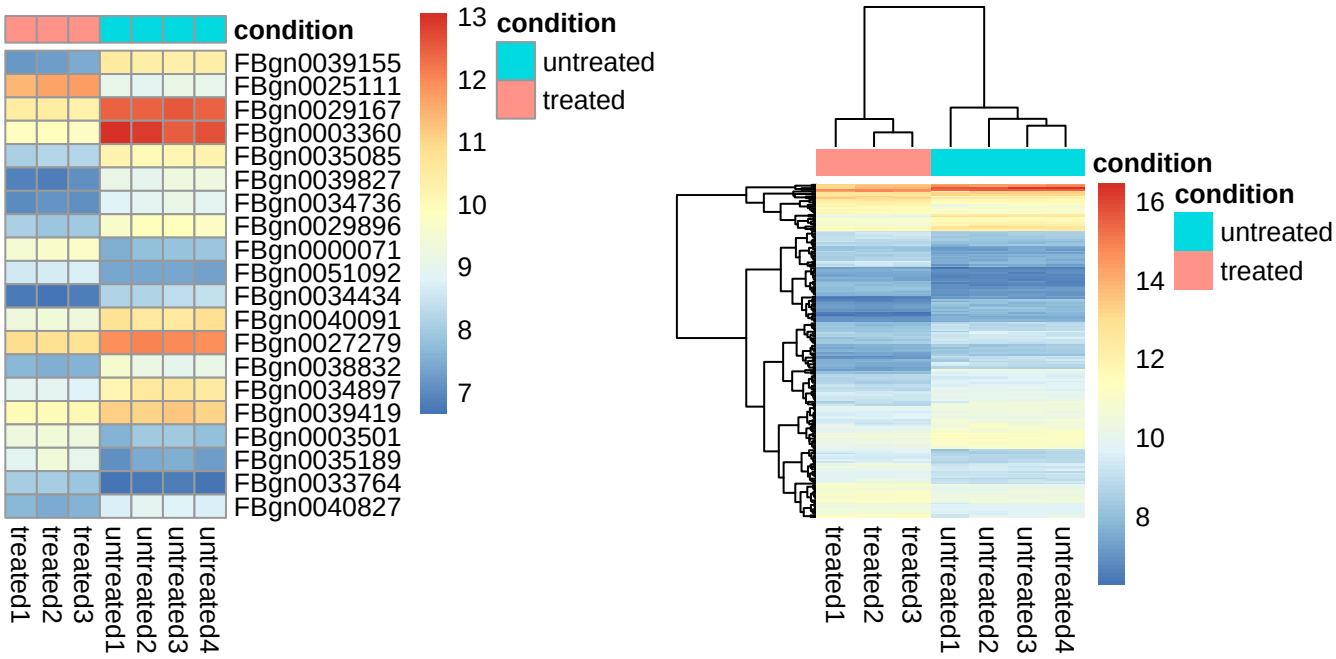
As part of our results output from DESeq2 we also provide the s-values. S-values provide an additional estimate of uncertainty by acting as a likelihood of whether the genes identified as differentially expressed are false positives. This helps our clients figure out if the DE genes they are seeing are worth confirming biologically. An s-value can be interpreted as s(genexyz)*100 = genexyz is x% likely to be a false positive finding. Here we obtain the s-values by performing log fold change (LFC) shrinkage on the dds object while specifying our comparison of interest and setting the s-values argument to TRUE. Briefly, LFC shrinkage makes the differences in the genes between groups comparable. Since genes can have a very small p-value even when the LFC is very small, this scales all of the LFCs by their p-values while preserving those truly large LFCs. We also use the apeglm estimator since it is the best available as of writing this and allows us to compute s-values. This is a much more complicated aspect of the analysis than we will cover here and for more detail on LFC shrinkage and s-values, please see the DESeq2 documentation here.

We check that the data frame was created successfully by using the informal unit test of dimension with the expected number of rows and columns and telling the program to stop if the file does not have these dimensions. After this runs successfully we would typically export them as a .csv file for you.

Visualizing

We now perform additional data visualizations. Typically we provide a PCA plot, heat maps, and volcano plots. We would be happy to work with you to customize these for publication. Please see our Data Visualization menu for more options and examples from previous projects.

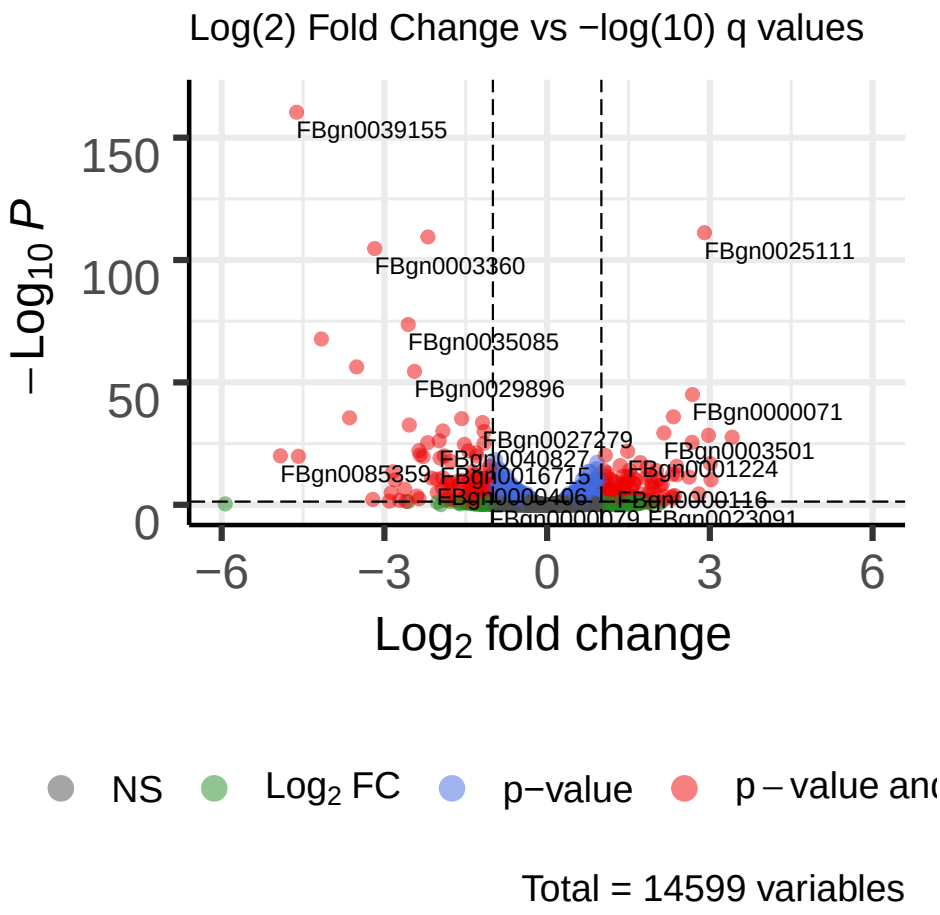
For our heat maps, we typically show the mean normalized counts of the samples. Next, we put the samples in the order of treated to untreated and select only the 20 genes with the lowest adjusted p-values be displayed so that we can see the gene names. It is possible to keep all DE genes on a heat map but after 20 it becomes nearly impossible to read each of the individual gene name and depending on the number of DE genes, it can be difficult to see the differences between groups with 100-200 genes on a plot setup this way. The first graph shows 20 genes and the second graph shows 200 genes.



Here we see the differences in mean-normalized counts between the samples in the treated vs untreated groups in the genes sorted by smallest adjusted p-value. Please note that these are sorted for convenience but the gene at the top of the list is no more significant than the gene at the bottom of the list. As is the case with nominal p-values, a smaller adjusted p-value does not make a gene more statistically significant than one with a larger adjusted p-value. If the genes are below the threshold, they are all equally statistically significantly differentially expressed. These are sorted for convenience but the gene at the top of the list is no more significant than the gene at the bottom of the list.

We also typically provide clients with an initial volcano plot created with the EnhancedVolcano R library. Similar to the PCA plot and heat map, this is a highly customizable graph and we would like to work with you to design graphs which best tell the story of your results.

A volcano plot is technically a scatter plot where the x-axis has the log2 transformed fold changes between the compared samples and the y axis has the local adjusted p-values for each gene, also called the q-value. Here we have also labelled the genes with FDR < 0.1 as that is where we set our threshold when we generated our results. The points in red are those which meet the threshold for statistical significance with a q value less than or equal to 0.1 and a log2 fold change of 1.0 or greater. Points in green are those with only log2 fold changes >1.0 and those in blue have claques < 0.1. The points in grey are non statistically significant by any measure. All of these parameters can be adjusted based on your cutoffs and thresholds.



There are many more functions and many more specifications to functions than are used here in order to show a simplified example of one of the tools we use for differential expression analysis. Obtaining specific, actionable, and publication quality results from analysis requires a deeper understanding of your specific data set and we would love the opportunity to discuss these options with you.

While we encourage clients to reach out prior to sequencing so that we can collaborate to design the experiment to answer your specific questions, we look forward to hearing from you at any stage of your RNA-seq project. Please find our contact information available on our website and check out some of the graphs we've made for previous clients here.

Session information and References

```
## [1] "Wed Mar 04 13:04:26 2020"
```

```
## R version 3.6.2 (2019-12-12)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 18363)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.1252
## [2] LC_CTYPE=English_United States.1252
```

```
## [3] LC_MONETARY=English_United States.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.1252
##
## attached base packages:
## [1] parallel  stats4    stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
##  [1] EnhancedVolcano_1.4.0       ggrepel_0.8.1
##  [3] pheatmap_1.0.12             ggplot2_3.2.1
##  [5] vsn_3.54.0                  DESeq2_1.26.0
##  [7] SummarizedExperiment_1.16.1 DelayedArray_0.12.2
##  [9] BiocParallel_1.20.1         matrixStats_0.55.0
## [11] Biobase_2.46.0              GenomicRanges_1.38.0
## [13] GenomeInfoDb_1.22.0         IRanges_2.20.2
## [15] S4Vectors_0.24.3            BiocGenerics_0.32.0
## [17] knitr_1.28                  dplyr_0.8.4
## [19] readr_1.3.1                 pacman_0.5.1
##
## loaded via a namespace (and not attached):
##  [1] bitops_1.0-6           bit64_0.9-7            RColorBrewer_1.1-2
##  [4] numDeriv_2016.8-1.1    tools_3.6.2            backports_1.1.5
##  [7] R6_2.4.1               affyio_1.56.0          rpart_4.1-15
## [10] Hmisc_4.3-1            DBI_1.1.0              lazyeval_0.2.2
## [13] colorspace_1.4-1       apeglm_1.8.0           nnet_7.3-12
## [16] withr_2.1.2            tidyselect_1.0.0       gridExtra_2.3
## [19] bit_1.1-15.2           compiler_3.6.2         preprocessCore_1.48.0
## [22] htmlTable_1.13.3       labeling_0.3           bookdown_0.17
## [25] scales_1.1.0           checkmate_2.0.0        mvtnorm_1.1-0
## [28] genefilter_1.68.0      affy_1.64.0            stringr_1.4.0
## [31] digest_0.6.25          foreign_0.8-72         rmarkdown_2.1
## [34] XVector_0.26.0         base64enc_0.1-3        jpeg_0.1-8.1
## [37] pkgconfig_2.0.3        htmltools_0.4.0        bbmle_1.0.23.1
## [40] limma_3.42.0           htmlwidgets_1.5.1      rlang_0.4.4
## [43] rstudioapi_0.11        RSQLite_2.2.0          farver_2.0.3
## [46] acepack_1.4.1          RCurl_1.98-1.1         magrittr_1.5
## [49] GenomeInfoDbData_1.2.2 Formula_1.2-3          Matrix_1.2-18
## [52] Rcpp_1.0.3             munsell_0.5.0          lifecycle_0.1.0
## [55] stringi_1.4.6          yaml_2.2.1             MASS_7.3-51.4
## [58] zlibbioc_1.32.0        plyr_1.8.6             grid_3.6.2
## [61] blob_1.2.1             bdsmatrix_1.3-4        crayon_1.3.4
## [64] lattice_0.20-38        splines_3.6.2          annotate_1.64.0
## [67] hms_0.5.3              locfit_1.5-9.1         pillar_1.4.3
## [70] geneplotter_1.64.0     XML_3.99-0.3           glue_1.3.1
## [73] evaluate_0.14          latticeExtra_0.6-29    data.table_1.12.8
## [76] BiocManager_1.30.10    png_0.1-7              vctrs_0.2.3
## [79] gtable_0.3.0           purrr_0.3.3            assertthat_0.2.1
## [82] emdbook_1.3.12         xfun_0.12              xtable_1.8-4
## [85] coda_0.19-3            survival_3.1-8         tibble_2.1.3
## [88] AnnotationDbi_1.48.0   memoise_1.1.0          cluster_2.1.0

##
## To cite the 'bookdown' package in publications use:
```

```
##
##   Yihui Xie (2020). bookdown: Authoring Books and Technical Documents
##   with R Markdown. R package version 0.17.
##
##   Yihui Xie (2016). bookdown: Authoring Books and Technical Documents
##   with R Markdown. Chapman and Hall/CRC. ISBN 978-1138700109
##
## To see these entries in BibTeX format, use 'print(<citation>,
## bibtex=TRUE)', 'toBibtex(.)', or set
## 'options(citation.bibtex.max=999)'.
```

```
##
## To cite package 'readr' in publications use:
##
##   Hadley Wickham, Jim Hester and Romain Francois (2018). readr: Read
##   Rectangular Text Data. R package version 1.3.1.
##   https://CRAN.R-project.org/package=readr
##
## A BibTeX entry for LaTeX users is
##
##   @Manual{,
##     title = {readr: Read Rectangular Text Data},
##     author = {Hadley Wickham and Jim Hester and Romain Francois},
##     year = {2018},
##     note = {R package version 1.3.1},
##     url = {https://CRAN.R-project.org/package=readr},
##   }
```

```
##
## To cite package 'dplyr' in publications use:
##
##   Hadley Wickham, Romain François, Lionel Henry and Kirill Müller
##   (2020). dplyr: A Grammar of Data Manipulation. R package version
##   0.8.4. https://CRAN.R-project.org/package=dplyr
##
## A BibTeX entry for LaTeX users is
##
##   @Manual{,
##     title = {dplyr: A Grammar of Data Manipulation},
##     author = {Hadley Wickham and Romain François and Lionel Henry and Kirill Müller},
##     year = {2020},
##     note = {R package version 0.8.4},
##     url = {https://CRAN.R-project.org/package=dplyr},
##   }
```

```
##
## To cite the 'knitr' package in publications use:
##
##   Yihui Xie (2020). knitr: A General-Purpose Package for Dynamic Report
##   Generation in R. R package version 1.28.
##
##   Yihui Xie (2015) Dynamic Documents with R and knitr. 2nd edition.
##   Chapman and Hall/CRC. ISBN 978-1498716963
```

```
##
##   Yihui Xie (2014) knitr: A Comprehensive Tool for Reproducible
##   Research in R. In Victoria Stodden, Friedrich Leisch and Roger D.
##   Peng, editors, Implementing Reproducible Computational Research.
##   Chapman and Hall/CRC. ISBN 978-1466561595
##
## To see these entries in BibTeX format, use 'print(<citation>,
## bibtex=TRUE)', 'toBibtex(.)', or set
## 'options(citation.bibtex.max=999)'.


##
## To cite ggplot2 in publications, please use:
##
##   H. Wickham. ggplot2: Elegant Graphics for Data Analysis.
##   Springer-Verlag New York, 2016.
##
## A BibTeX entry for LaTeX users is
##
##   @Book{,
##     author = {Hadley Wickham},
##     title = {ggplot2: Elegant Graphics for Data Analysis},
##     publisher = {Springer-Verlag New York},
##     year = {2016},
##     isbn = {978-3-319-24277-4},
##     url = {https://ggplot2.tidyverse.org},
##   }


##
##   Love, M.I., Huber, W., Anders, S. Moderated estimation of fold change
##   and dispersion for RNA-seq data with DESeq2 Genome Biology 15(12):550
##   (2014)
##
## A BibTeX entry for LaTeX users is
##
##   @Article{,
##     title = {Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2},
##     author = {Michael I. Love and Wolfgang Huber and Simon Anders},
##     year = {2014},
##     journal = {Genome Biology},
##     doi = {10.1186/s13059-014-0550-8},
##     volume = {15},
##     issue = {12},
##     pages = {550},
##   }


##
## To cite the vsn package in publications use:
##
##   Wolfgang Huber, Anja von Heydebreck, Holger Sueltmann, Annemarie
##   Poustka and Martin Vingron. Variance Stabilization Applied to
##   Microarray Data Calibration and to the Quantification of Differential
##   Expression. Bioinformatics 18, S96-S104 (2002).
##
```

```
## A BibTeX entry for LaTeX users is
##
##   @Article{,
##     title = {Variance Stabilization Applied to Microarray Data Calibration and to the Quantification
##     author = {Wolfgang Huber and Anja {von Heydebreck} and Holger Sueltmann and Annemarie Poustka an
##     journal = {Bioinformatics},
##     year = {2002},
##     volume = {18 Suppl. 1},
##     pages = {S96-S104},
##   }

##
## To cite package 'pheatmap' in publications use:
##
##   Raivo Kolde (2019). pheatmap: Pretty Heatmaps. R package version
##   1.0.12. https://CRAN.R-project.org/package=pheatmap
##
## A BibTeX entry for LaTeX users is
##
##   @Manual{,
##     title = {pheatmap: Pretty Heatmaps},
##     author = {Raivo Kolde},
##     year = {2019},
##     note = {R package version 1.0.12},
##     url = {https://CRAN.R-project.org/package=pheatmap},
##   }
##
## ATTENTION: This citation information has been auto-generated from the
## package DESCRIPTION file and may need manual editing, see
## 'help("citation")'.

##
## To cite package 'EnhancedVolcano' in publications use:
##
##   Kevin Blighe, Sharmila Rana and Myles Lewis (2019). EnhancedVolcano:
##   Publication-ready volcano plots with enhanced colouring and labeling.
##   R package version 1.4.0.
##   https://github.com/kevinblighe/EnhancedVolcano
##
## A BibTeX entry for LaTeX users is
##
##   @Manual{,
##     title = {EnhancedVolcano: Publication-ready volcano plots with enhanced colouring and
## labeling},
##     author = {Kevin Blighe and Sharmila Rana and Myles Lewis},
##     year = {2019},
##     note = {R package version 1.4.0},
##     url = {https://github.com/kevinblighe/EnhancedVolcano},
##   }
##
## ATTENTION: This citation information has been auto-generated from the
## package DESCRIPTION file and may need manual editing, see
## 'help("citation")'.
```