Day 7 - Introduction to Gene Differential Expression Analysis using DESeq2

Authors: Jacob Stanley (jacob.stanley@colorado.edu). Edited and updated by Daniel Ramirez (daniel.ramirezhernandez@colorado.edu).

Additional DESeq2 resources:
https://bioconductor.org/packages/release/bioc/html/DESeq2.html
https://bioconductor.org/packages/release/bioc/vignettes/DESeq2/inst/doc/DESeq2.html

On Day 6 you learned how to use the software featureCounts to obtain a text file containing the number of reads mapping to each gene from a given annotation file. Today on Day 7, we will use those gene counts tables as input for the software DESeq2 to answer the question: What genes are statistically significantly changed upon an experimental condition? In particular, we will explore a dataset from a real experiment I am analyzing. You will use a gene count table that we already prepared for you, from an experiment where Owl Monkey (Aotus nancymaae) lymphoblastoid cells were treated with either the vehicle DMSO, or with the p53-activator drug Nutlin.

The purpose of DESeq2 is to identify which genomic loci demonstrate a statistically significant difference in expression level between two or more conditions (referred to as “gene differential expression analysis”). It does so by modeling the variance in expression level across the full range of baseline expression levels present in the data, and determines if the differential expression level for each loci is significantly greater than this variance. DESeq2 takes as an input the unnormalized count values for each (non-overlapping) loci in each sample. We recommend that you use featureCounts() to compute count values. DESeq2 performs best when provided multiple replicates per experimental condition (preferably 5+ replicates), in order to get an accurate estimation of within condition variance. DESeq2 is only to be used for non-overlapping, unique genomic loci. If one’s aim is to compute differential expression of transcripts, DESeq2 is not appropriate.

Note: All commands are executed within the R environment. We will be executing them manually, from the R command line, but they can also be compiled into a single script to be executed together.

--- COPYING SCRATCH FILES TO LOCAL MACHINE ---

Go to the general/communal workshop scratch directory for Day 7 and copy these featureCount output files as well as the condition table file we used to run featureCounts onto your own local machine (do not copy the files onto your scratch users directory). We will be using your local machine installation of RStudio and use these files as inputs.

/scratch/Shares/public/sread2022/data_files/day7

RNA-OwlMonkey-Nutlin.featureCounts.annotation.tsv
RNA-OwlMonkey-Nutlin.featureCounts.data.coverage.tsv
RNA-OwlMonkey-Nutlin.featureCounts.stat.tsv
RNA-OwlMonkey-Nutlin.featureCounts.targets.tsv
conditionsTable.RNA-Nutlin-OwlMonkey.csv

Shown below is the top of the file **RNA-OwlMonkey-Nutlin.featureCounts.data.coverage.tsv**, showing the raw unnormalized read counts across the first genes of the annotation file.

```
X.scratch.Users.dara6367.SR2022.day7.bam.RNA.DMSO.OwlMonkey.1.Anan_20.sorted.bam        X.s
cratch.Users.dara6367.SR2022.day7.bam.RNA.DMSO.OwlMonkey.2.Anan_20.sorted.bam  X.scratch.Us
ers.dara6367.SR2022.day7.bam.RNA.DMSO.OwlMonkey.3.Anan_20.sorted.bam        X.scratch.Users
.dara6367.SR2022.day7.bam.RNA.Nutlin.OwlMonkey.1.Anan_20.sorted.bam        X.scratch.Users.
dara6367.SR2022.day7.bam.RNA.Nutlin.OwlMonkey.2.Anan_20.sorted.bam      X.scratch.Users.dar
a6367.SR2022.day7.bam.RNA.Nutlin.OwlMonkey.3.Anan_20.sorted.bam
KIF4B   0       2       0       0       0       1
MRPL22  1282    1490    1119    1499    1391    1522
GEMIN5  2135    2533    2115    1929    1720    1914
CNOT8   1244    1479    1427    1659    1272    1236
LOC105715937    140     160     126     152     90      117
FAXDC2  77      66      69      344     416     335
LARP1   16998   18534   14562   16149   12879   14337
LOC110567054    0       0       1       0       0       0
LOC105715318    5       4       3       2       8       7
```

Shown below is the contents of the **conditionsTable.RNA-Nutlin-OwlMonkey.csv** file. It has four columns, each with information regarding each of the RNA-seq datasets. This information was used by featureCounts, and will be used again by DESeq2 to know which datasets are replicates of each other, and against what other replicates to make any comparison.

```
bamFileName,sample,species,treatment
RNA-DMSO-OwlMonkey-1.Anan_20.sorted.bam,RNA-DMSO-OwlMonkey-1,OwlMonkey,DMSO
RNA-DMSO-OwlMonkey-2.Anan_20.sorted.bam,RNA-DMSO-OwlMonkey-2,OwlMonkey,DMSO
RNA-DMSO-OwlMonkey-3.Anan_20.sorted.bam,RNA-DMSO-OwlMonkey-3,OwlMonkey,DMSO
RNA-Nutlin-OwlMonkey-1.Anan_20.sorted.bam,RNA-Nutlin-OwlMonkey-1,OwlMonkey,Nutlin
RNA-Nutlin-OwlMonkey-2.Anan_20.sorted.bam,RNA-Nutlin-OwlMonkey-2,OwlMonkey,Nutlin
RNA-Nutlin-OwlMonkey-3.Anan_20.sorted.bam,RNA-Nutlin-OwlMonkey-3,OwlMonkey,Nutlin
```

--- USING DESEQ2 WITHIN RSTUDIO ---

We will use DESeq2 with the RStudio on your local computer. First, we need to tell RStudio to load the DESeq2 library (if you have not yet installed DESeq2, let a class helper know. Be aware though that installing DESeq2 takes some time as it also needs several dependencies installed as well). Loading DESeq2 will prompt some red messages, and they should end with R normal “greater than” prompt symbol.

```
> library(DESeq2)
Loading required package: S4Vectors
Loading required package: stats4
Loading required package: BiocGenerics
Loading required package: parallel

Attaching package: 'BiocGenerics'

The following objects are masked from 'package:parallel':
```

Load the two input files:

1) Load the conditions table, and rename its row names with the “sample” column.

```
conditionsTableFile <-
"/home/daniel/Downloads/conditionsTable.RNA-Nutlin-OwlMonkey.csv"

conditionsTable <- read.csv(conditionsTableFile, header = TRUE)

rownames(conditionsTable) <- conditionsTable$sample
```

```
> conditionsTable
                              bamFileName                 sample   species treatment
1   RNA-DMSO-OwlMonkey-1.Anan_20.sorted.bam   RNA-DMSO-OwlMonkey-1 OwlMonkey      DMSO
2   RNA-DMSO-OwlMonkey-2.Anan_20.sorted.bam   RNA-DMSO-OwlMonkey-2 OwlMonkey      DMSO
3   RNA-DMSO-OwlMonkey-3.Anan_20.sorted.bam   RNA-DMSO-OwlMonkey-3 OwlMonkey      DMSO
4 RNA-Nutlin-OwlMonkey-1.Anan_20.sorted.bam RNA-Nutlin-OwlMonkey-1 OwlMonkey    Nutlin
5 RNA-Nutlin-OwlMonkey-2.Anan_20.sorted.bam RNA-Nutlin-OwlMonkey-2 OwlMonkey    Nutlin
6 RNA-Nutlin-OwlMonkey-3.Anan_20.sorted.bam RNA-Nutlin-OwlMonkey-3 OwlMonkey    Nutlin
```

2) Load the raw gene counts table, and rename its column names with the “sample” column from the conditions table.

```
geneCountsTableFile <-
"/home/daniel/Downloads/RNA_OwlMonkey_Nutlin.featureCounts.data.coverage
.tsv"

geneCountsTable <- read.table(geneCountsTableFile,
                              header = TRUE, sep = "\t", fill = TRUE,
                              stringsAsFactors = FALSE, na.strings = "")

colnames(geneCountsTable) <- conditionsTable$sample
```

```
> geneCountsTable
            RNA-DMSO-OwlMonkey-1 RNA-DMSO-OwlMonkey-2 RNA-DMSO-OwlMonkey-3 RNA-Nutlin-OwlMonkey-1 RNA-Nutlin-OwlMonkey-2 RNA-Nutlin-OwlMonkey-3
KIF4B                          0                    2                    0                     0                     0                     1
MRPL22                      1282                 1490                 1119                  1499                  1391                  1522
GEMIN5                      2135                 2533                 2115                  1929                  1720                  1914
CNOT8                       1244                 1479                 1427                  1659                  1272                  1236
LOC105715937                 140                  160                  126                   152                    90                   117
FAXDC2                        77                   66                   69                   344                   416                   335
LARP1                      16998                18534                14562                 16149                 12879                 14337
LOC110567054                   0                    0                    1                     0                     0                     0
LOC105715318                   5                    4                    3                     2                     8                     7
LOC110567052                  17                   37                   18                    28                    37                    49
LOC105715327                   0                    0                    0                     0                     0                     0
HAND1                          0                    1                    0                     0                     0                     0
SAP30L                       399                  401                  350                   363                   281                   343
```

Next, load the two inputs onto DESeq2 using the following function. You can then type the variable **dds** and see some information of its contents, including its variable type, the number of genes that have counts (e.g. 31,324), and some of the gene and datasets labels.

```
dds <- DESeqDataSetFromMatrix(countData = geneCountsTable,
                              colData = conditionsTable,
                              design = ~ treatment)
```

```
> dds <- DESeqDataSetFromMatrix(countData = geneCountsTable,
+                               colData = conditionsTable,
+                               design = ~ treatment)
> dds
class: DESeqDataSet
dim: 31324 6
metadata(1): version
assays(1): counts
rownames(31324): KIF4B MRPL22 ... LOC105714205 LOC105719629
rowData names(0):
colnames(6): RNA-DMSO-OwlMonkey-1 RNA-DMSO-OwlMonkey-2 ... RNA-Nutlin-OwlMonkey-2 RNA-Nutlin-OwlMonkey-3
colData names(4): bamFileName sample species treatment
```

Optionally, you can remove all the gene entries that have low counts, such as those gene entries that have mostly zero counts. If you do not remove them, DESeq2 will automatically remove them internally while doing its calculations. Notice that printing the **dds** variable again gives us a smaller number of genes with counts (e.g. 25,344).

```
dds <- dds[rowSums(counts(dds)) > 1,]
```

```
> dds <- dds[rowSums(counts(dds)) > 1,]
> dds
class: DESeqDataSet
dim: 25344 6
metadata(1): version
assays(1): counts
rownames(25344): KIF4B MRPL22 ... ND6 CYTB
rowData names(0):
colnames(6): RNA-DMSO-OwlMonkey-1 RNA-DMSO-OwlMonkey-2 ... RNA-Nutlin-OwlMonkey-2 RNA-Nutlin-OwlMonkey-3
colData names(4): bamFileName sample species treatment
```

Run DESeq2's main function **DESeq** on the **dds** variable you created. DESeq2 will internally do several actions: it will estimate each dataset size scale factors, it will estimate dispersion, it will fit a generalized linear model, it will calculate each gene's fold change..

```
DEdds <- DESeq(dds)
```

```
> DEdds <- DESeq(dds)
estimating size factors
estimating dispersions
gene-wise dispersion estimates
mean-dispersion relationship
final dispersion estimates
fitting model and testing
```

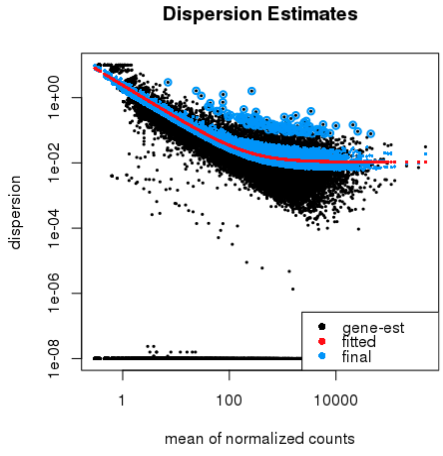
Check what size factors were estimated for each of the 6 Owl Monkey datasets. Check how the total number of gene-assigned reads changes to a more homogeneous number with the normalization (e.g. between 25 and 25 million counts).

```
sizeFactors(DEdds)
colSums(counts(DEdds, normalized = FALSE))
colSums(counts(DEdds, normalized = TRUE))
```

```
> sizeFactors(DEdds)
  RNA-DMSO-OwlMonkey-1   RNA-DMSO-OwlMonkey-2   RNA-DMSO-OwlMonkey-3 RNA-Nutlin-OwlMonkey-1 RNA-Nutlin-OwlMonkey-2 RNA-Nutlin-OwlMonkey-3
             0.9099632              1.1133175              0.8814280             1.0989803             0.9589769             1.0838659
> colSums(counts(DEdds, normalized = FALSE))
  RNA-DMSO-OwlMonkey-1   RNA-DMSO-OwlMonkey-2   RNA-DMSO-OwlMonkey-3 RNA-Nutlin-OwlMonkey-1 RNA-Nutlin-OwlMonkey-2 RNA-Nutlin-OwlMonkey-3
              23558049               28103457               22961238              28175764              24217229              27407655
> colSums(counts(DEdds, normalized = TRUE))
  RNA-DMSO-OwlMonkey-1   RNA-DMSO-OwlMonkey-2   RNA-DMSO-OwlMonkey-3 RNA-Nutlin-OwlMonkey-1 RNA-Nutlin-OwlMonkey-2 RNA-Nutlin-OwlMonkey-3
              25889013               25242985               26050043              25638098              25253193              25286944
```

You can check the dispersion estimates with a simple DESeq2 function. You want to see that the estimates are monotonically descending, and that most data points (blue) are nearby the fitted line (red).

```
plotDispEsts(DEdds, main = "Dispersion Estimates")
```



Define the alpha value that DESeq2 will need to assign statistical significance, as well as the names of the two experimental conditions we want to compare against each other.

```
alphaValue <- 0.05
contrast <- c("treatment", "Nutlin", "DMSO")
```

Extract statistically significant results, and do DESeq2 special log fold-change shrinkage, which is useful for visualization purposes. DESeq2 will let you know that it is using the normal algorithm for doing the shrinkage, and that there are newer algorithms if you want to test them out as well. They require independent library installation.

```
results <- results(DEdds, alpha = alphaValue, contrast = contrast)
results_shrunk <- lfcShrink(DEdds, contrast = contrast, res = results)
```

```
> results <- results(DEdds, alpha = alphaValue, contrast = contrast)
> results_shrunk <- lfcShrink(DEdds, contrast = contrast, res = results)
using 'normal' for LFC shrinkage, the Normal prior from Love et al (2014).

Note that type='apeglm' and type='ashr' have shown to have less bias than type='normal'
See ?lfcShrink for more details on shrinkage type, and the DESeq2 vignette.
Reference: https://doi.org/10.1093/bioinformatics/bty895
```

See that both unshrunk and shrunk results have identical nominal and adjusted p-values. The shrinkage only affects the fold-change estimation values.

```
> results
log2 fold change (MLE): treatment Nutlin vs DMSO
Wald test p-value: treatment Nutlin vs DMSO
DataFrame with 25344 rows and 6 columns
                    baseMean      log2FoldChange             lfcSE                stat              pvalue               padj
                   <numeric>           <numeric>         <numeric>           <numeric>           <numeric>          <numeric>
KIF4B        0.453176014417389  -0.853669839528624   3.1459092765117  -0.271358696165328   0.786115166265598                 NA
MRPL22        1372.57499728445  0.0705568974490746 0.110393226783761   0.639141544320305   0.522730855753109   0.84390159312979
GEMIN5        2055.94813146002  -0.401543864766333 0.102081544153861   -3.93355986231078 8.3696982120567e-05 0.00188921892160654
CNOT8         1381.81187840213  -0.117229808511939 0.138913240640311     -0.843906656929   0.398721567991109   0.764846515184746
LOC105715937  130.103970717827  -0.369854735970895 0.211309382715303     -1.7502996375187   0.080066623438432   0.348832028118737
...                        ...                 ...               ...                 ...                 ...                 ...
ND4L          14622.6342126516  -0.139877638084475 0.177878526086521   -0.786366073308015   0.431653053487173   0.787894022336803
ND4           108267.241892733  -0.122639601082198 0.164439221758208   -0.745805044386112   0.455785192304676   0.804825210617017
ND5           94874.1849046641 -0.00814286793521667 0.164408998074295 -0.0495281160434844   0.960498431326764    0.9940027830031
ND6           11473.7550456541  -0.0250797106720452 0.179396645056726   -0.139800332743764   0.888817750386421   0.977085671775118
CYTB          47599.7978520119   -0.155087478728792  0.15138397148572    -1.02446432873127   0.305616011089138   0.689761476549421
```

```
> results_shrunk
log2 fold change (MAP): treatment Nutlin vs DMSO
Wald test p-value: treatment Nutlin vs DMSO
DataFrame with 25344 rows and 6 columns
                    baseMean      log2FoldChange             lfcSE                stat              pvalue               padj
                   <numeric>           <numeric>         <numeric>           <numeric>           <numeric>          <numeric>
KIF4B        0.453176014417389 -0.0205657480835733 0.0663424443325526  -0.271358696165328   0.786115166265598                 NA
MRPL22        1372.57499728445  0.0667663124748174  0.104462375386229   0.639141544320305   0.522730855753109   0.84390159312979
GEMIN5        2055.94813146002  -0.382952180076924 0.0973535752685086   -3.93355986231078 8.3696982120567e-05 0.00188921892160654
CNOT8         1381.81187840213  -0.107559528790941  0.127455741491742     -0.843906656929   0.398721567991109   0.764846515184746
LOC105715937  130.103970717827   -0.30620680945027  0.174892945496475     -1.7502996375187   0.080066623438432   0.348832028118737
...                        ...                 ...               ...                 ...                 ...                 ...
ND4L          14622.6342126516  -0.121908432391056  0.155027765294118   -0.786366073308015   0.431653053487173   0.787894022336803
ND4           108267.241892733  -0.108919510026877  0.146042739336064   -0.745805044386112   0.455785192304676   0.804825210617017
ND5           94874.1849046641 -0.00723238363776809   0.14602190179534 -0.0495281160434844   0.960498431326764    0.9940027830031
ND6           11473.7550456541  -0.0218099389282525  0.156007332220667   -0.139800332743764   0.888817750386421   0.977085671775118
CYTB          47599.7978520119   -0.140127648233436  0.136781350849677    -1.02446432873127   0.305616011089138   0.689761476549421
```

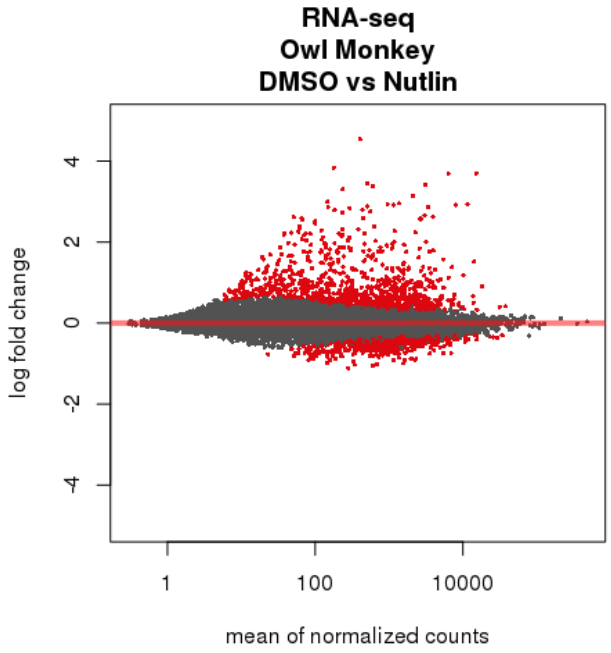
You can check a global visualization of how the Owl Monkey genes changed upon the Nutlin treatment using DESeq2 **plotMA** function.

Notice that you can call a given library's function by calling the name of the library followed by two colons and the name of the function. This helps R in case there are two functions with conflicting names.

The resulting MA figure will plot each gene's fold-change in the Y-axis, and such gene's normalized counts in the X-axis. You can tell **plotMA** to color genes (red are significant, gray are non-significant) by significance by using the same alpha level threshold you defined previously.

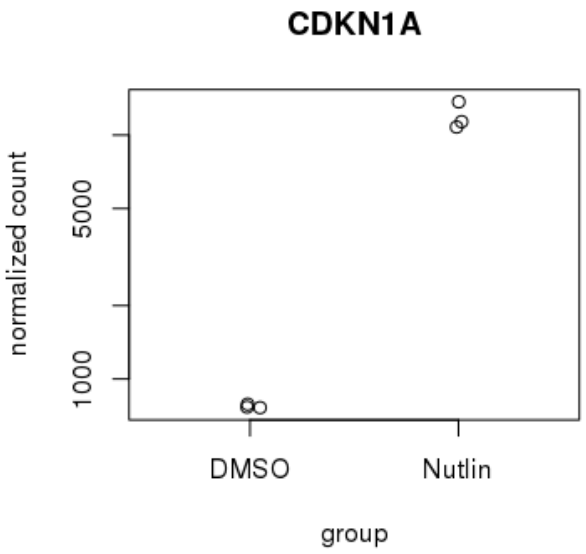
You can observe that there are more red dots with positive than negative fold-change. This behavior will depend on the treatment that the cells of your experiment are exposed to.

```
DESeq2::plotMA(results_shrunk,
        alpha = alphaValue,
        main = "RNA-seq\nOwl Monkey\nDMSO vs Nutlin",
        xlab = "mean of normalized counts",
        ylab = "log fold change",
        ylim = c(-5,5))
```



You can plot the normalized read counts for a given gene using DESeq2 built-in function **plotCounts**. You need to tell it the name of the gene as it appears in the original annotation file, as well as the name of the column in the condition file that denotes either "Nutlin" or "DMSO" as the conditions.

```
gene <- "CDKN1A"
plotCounts(DEdds, gene,
        intgroup = "treatment",
        normalized = TRUE)
```



Order the gene results in descending order by their adjusted p-values, so that the most significant genes will be on the top of your results table. You can see that the very top gene is CDKN1A or p21, a known gene that controls cell cycle progression directly controlled by the transcription factor p53, which itself is activated by the drug Nutlin.

```
results_shrunk <- results_shrunk[order(results_shrunk$padj), ]
```

```
> results_shrunk <- results_shrunk[order(results_shrunk$padj), ]
> results_shrunk
log2 fold change (MAP): treatment Nutlin vs DMSO
Wald test p-value: treatment Nutlin vs DMSO
DataFrame with 25344 rows and 6 columns
                    baseMean       log2FoldChange             lfcSE                stat              pvalue               padj
                   <numeric>            <numeric>         <numeric>           <numeric>           <numeric>          <numeric>
CDKN1A        6358.40143493648      3.69206893076486  0.114185141989414     32.2600135327618 2.54615964984832e-228 4.70122917747994e-224
LOC105711961  11550.4918297821      2.92377991177745  0.096543209826277     30.2682798249532 2.99888851552039e-201 2.76857387752843e-197
TRIM35        15064.0262681737       3.6969107859406  0.122600505324706     30.1315182422385 1.87343979599633e-199 1.15303974644254e-195
LOC105711962  8096.25520005702      2.91606487719087  0.102525297049144     28.4203629824931 1.13305093366821e-177 5.23016310981245e-174
BBC3          3098.11169800706      3.40849722483483  0.128413884263962      26.446588058405 3.99383200278336e-154 1.47484228198784e-150
...                        ...                   ...                ...                 ...                 ...                ...
SKOR1         2.32578538035236     0.230511193796099  0.142587216697769     1.67912585752502   0.0931275177784614                 NA
LOC105716603  3.48925176406827    -0.174169099305899  0.170784985544508    -1.02801836806179    0.303941187534891                 NA
LOC105716644 0.338789820432744   -0.0352449354212874 0.0603415368072825   -0.564948030467136    0.572109113122174                 NA
LOC105716635  2.15546088102312    0.0623038332212818  0.137816230883515     0.454691164746993    0.649331438055529                 NA
LOC105716616 0.686382478138873 -0.000291539476904465 0.0820318771350455 -0.00358390973476043    0.997140459876842                 NA
```

Filter out only those genes whose adjusted p-value are less than the defined alpha value. Finally, store your significant genes onto a text output file. You can use this file for any downstream analysis you deem appropriate, including gene set enrichment software such as GSEA and others.

```
results_shrunk_sig <- subset(results_shrunk, padj < alphaValue)

write.table(results_shrunk_sig, sep = "\t", quote = FALSE, row.names = TRUE,
col.names = TRUE,
"/home/daniel/Downloads//DESeq2_RNA-seq_OwlMonkey_Nutlin_results.tsv")
```

You can then see the output text file from your bash terminal.

```
head DESeq2_RNA-seq_OwlMonkey_Nutlin_results.tsv | cut -f1,2,7
```

```
baseMean        log2FoldChange
CDKN1A  6358.40143493648        4.70122917747994e-224
LOC105711961    11550.4918297821        2.76857387752843e-197
TRIM35  15064.0262681737        1.15303974644254e-195
LOC105711962    8096.25520005702        5.23016310981245e-174
BBC3    3098.11169800706        1.47484228198784e-150
EDA2R   2581.3081015414 5.13331645923917e-127
SULF2   3038.72217109827        4.07278968751389e-117
ABHD4   3382.86888055437        1.07404131042245e-115
NATD1   1368.54635485781        6.67104561070011e-115
```