**CODES**

**Jupyther Notebook Initialization**

*# Import required python module that enables command-line tools through '%%bash' and R code through '%%R' tags.*

**import** **rpy2**

%**load\_ext** rpy2.ipython

*# Import R packages to rpy2*

**import** **rpy2.robjects** **as** **robjects**

**import** **rpy2.robjects.packages** **as** **rpackages**

**from** **rpy2.robjects.packages** **import** importr

*# Prevent unnecessary warnings from loading R packages*

**import** **warnings**

warnings.filterwarnings("ignore", category=**RuntimeWarning**)

*# Utilize full browser windown width*

**from** **IPython.core.display** **import** display, HTML

display(HTML("<style>.container { width:100% !important; }</style>"))

display(HTML("<style>pre {white-space: pre; }</style>"))

%**env** ProjectFolder = /home/jupyterAbiVatsa

%**env** RawDataFolder = /home/jupyterAbiVatsa/bigdata/ProjectWork\_sequences/set1

**Analysis**

%%bash

*# List .fastq files*

ls /home/jupyterAbiVatsa/bigdata/ProjectWork\_sequences/set1

dexa\_2h\_rep1.MDAMB231.SRR2046034\_2M.fastq

dexa\_2h\_rep2.MDAMB231.SRR2046035\_2M.fastq

dexa\_4h\_rep1.MDAMB231.SRR2046036\_2M.fastq

dexa\_4h\_rep2.MDAMB231.SRR2046037\_2M.fastq

vehi\_NAh\_rep1.MDAMB231.SRR2046032\_2M.fastq

vehi\_NAh\_rep2.MDAMB231.SRR2046033\_2M.fastq

%%bash

*# View the contents of .fastq file*

head -8 /home/jupyterAbiVatsa/bigdata/ProjectWork\_sequences/set1/dexa\_2h\_rep1.MDAMB231.SRR2046034\_2M.fastq

@SRR2046034.40838091 SN1052:291:C6DAJACXX:1:2110:12419:78282 length=50

GTTGAGACCAAGTATGATTGACATAATCATTCATTTCGACCCCTCCTGCC

+

BBBFFFFFGHHHHGJJGIJIFHH@HC>DEFFGIIIIGIGGGHGGGIJIBB

@SRR2046034.57624042 SN1052:291:C6DAJACXX:1:2301:18710:4362 length=50

CACTGGCAGTGGCAATGCTGGCCGTCTGGCAACTTTTCGTCTCCCCGCGG

+

CCCFFFFFHHHHHJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJJH

%%bash

*# Count the number of lines in each .fastq file*

wc -l /home/jupyterAbiVatsa/bigdata/ProjectWork\_sequences/set1/\*.fastq

8000000 /home/jupyterAbiVatsa/bigdata/ProjectWork\_sequences/set1/dexa\_2h\_rep1.MDAMB231.SRR2046034\_2M.fastq

8000000 /home/jupyterAbiVatsa/bigdata/ProjectWork\_sequences/set1/dexa\_2h\_rep2.MDAMB231.SRR2046035\_2M.fastq

8000000 /home/jupyterAbiVatsa/bigdata/ProjectWork\_sequences/set1/dexa\_4h\_rep1.MDAMB231.SRR2046036\_2M.fastq

8000000 /home/jupyterAbiVatsa/bigdata/ProjectWork\_sequences/set1/dexa\_4h\_rep2.MDAMB231.SRR2046037\_2M.fastq

8000000 /home/jupyterAbiVatsa/bigdata/ProjectWork\_sequences/set1/vehi\_NAh\_rep1.MDAMB231.SRR2046032\_2M.fastq

8000000 /home/jupyterAbiVatsa/bigdata/ProjectWork\_sequences/set1/vehi\_NAh\_rep2.MDAMB231.SRR2046033\_2M.fastq

48000000 total

### **Raw read quality control**

In [5]:

%%bash

mkdir -p "**${**ProjectFolder**}**/FastQC\_initial"

files=**$(**find "**${**RawDataFolder**}**" | grep ".fastq$"**)**

fastqc **\**

$files **\**

-t 1 **\**

-o "**${**ProjectFolder**}**/FastQC\_initial" **\**

--quiet

echo "All done!"

All done!

### **Explore the .html FastQC reports for the raw data**

### **Read trimming and quality control**

\*\*NOTE:\*\* Supply appropriate values to the parameters for Trimmomatic before running it. See Practical1 notebook for guidance. Additional help for using Trimmomatic: <http://www.usadellab.org/cms/?page=trimmomatic>.

In [6]:

%%bash

mkdir -p "**${**ProjectFolder**}**/trimmed\_reads"

**for** file in **$(**ls **${**RawDataFolder**}** | grep ".fastq$"**)**; **do**

filename=`echo **${**file**}** | sed 's/.fastq/.trimmed.fastq/'`

outFile="**${**ProjectFolder**}**/trimmed\_reads/**${**filename**}**"

*# Set parameters*

TrimmomaticSE **\**

-phred33 **\**

"**${**RawDataFolder**}**/**${**file**}**" **\**

"**${**outFile**}**" **\**

ILLUMINACLIP:TruSeq3-SE.fa:2:30:10 **\**

LEADING:3 **\**

TRAILING:3 **\**

SLIDINGWINDOW:4:15 **\**

MINLEN:33

**done**

TrimmomaticSE: Started with arguments:

-phred33 /home/jupyterAbiVatsa/bigdata/ProjectWork\_sequences/set1/dexa\_2h\_rep1.MDAMB231.SRR2046034\_2M.fastq /home/jupyterAbiVatsa/trimmed\_reads/dexa\_2h\_rep1.MDAMB231.SRR2046034\_2M.trimmed.fastq ILLUMINACLIP:TruSeq3-SE.fa:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:33

Automatically using 1 threads

Using Long Clipping Sequence: 'AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGTA'

Using Long Clipping Sequence: 'AGATCGGAAGAGCACACGTCTGAACTCCAGTCAC'

ILLUMINACLIP: Using 0 prefix pairs, 2 forward/reverse sequences, 0 forward only sequences, 0 reverse only sequences

Input Reads: 2000000 Surviving: 1983162 (99.16%) Dropped: 16838 (0.84%)

TrimmomaticSE: Completed successfully

TrimmomaticSE: Started with arguments:

-phred33 /home/jupyterAbiVatsa/bigdata/ProjectWork\_sequences/set1/dexa\_2h\_rep2.MDAMB231.SRR2046035\_2M.fastq /home/jupyterAbiVatsa/trimmed\_reads/dexa\_2h\_rep2.MDAMB231.SRR2046035\_2M.trimmed.fastq ILLUMINACLIP:TruSeq3-SE.fa:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:33

Automatically using 1 threads

Using Long Clipping Sequence: 'AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGTA'

Using Long Clipping Sequence: 'AGATCGGAAGAGCACACGTCTGAACTCCAGTCAC'

ILLUMINACLIP: Using 0 prefix pairs, 2 forward/reverse sequences, 0 forward only sequences, 0 reverse only sequences

Input Reads: 2000000 Surviving: 1688880 (84.44%) Dropped: 311120 (15.56%)

TrimmomaticSE: Completed successfully

TrimmomaticSE: Started with arguments:

-phred33 /home/jupyterAbiVatsa/bigdata/ProjectWork\_sequences/set1/dexa\_4h\_rep1.MDAMB231.SRR2046036\_2M.fastq /home/jupyterAbiVatsa/trimmed\_reads/dexa\_4h\_rep1.MDAMB231.SRR2046036\_2M.trimmed.fastq ILLUMINACLIP:TruSeq3-SE.fa:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:33

Automatically using 1 threads

Using Long Clipping Sequence: 'AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGTA'

Using Long Clipping Sequence: 'AGATCGGAAGAGCACACGTCTGAACTCCAGTCAC'

ILLUMINACLIP: Using 0 prefix pairs, 2 forward/reverse sequences, 0 forward only sequences, 0 reverse only sequences

Input Reads: 2000000 Surviving: 1975678 (98.78%) Dropped: 24322 (1.22%)

TrimmomaticSE: Completed successfully

TrimmomaticSE: Started with arguments:

-phred33 /home/jupyterAbiVatsa/bigdata/ProjectWork\_sequences/set1/dexa\_4h\_rep2.MDAMB231.SRR2046037\_2M.fastq /home/jupyterAbiVatsa/trimmed\_reads/dexa\_4h\_rep2.MDAMB231.SRR2046037\_2M.trimmed.fastq ILLUMINACLIP:TruSeq3-SE.fa:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:33

Automatically using 1 threads

Using Long Clipping Sequence: 'AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGTA'

Using Long Clipping Sequence: 'AGATCGGAAGAGCACACGTCTGAACTCCAGTCAC'

ILLUMINACLIP: Using 0 prefix pairs, 2 forward/reverse sequences, 0 forward only sequences, 0 reverse only sequences

Input Reads: 2000000 Surviving: 1684798 (84.24%) Dropped: 315202 (15.76%)

TrimmomaticSE: Completed successfully

TrimmomaticSE: Started with arguments:

-phred33 /home/jupyterAbiVatsa/bigdata/ProjectWork\_sequences/set1/vehi\_NAh\_rep1.MDAMB231.SRR2046032\_2M.fastq /home/jupyterAbiVatsa/trimmed\_reads/vehi\_NAh\_rep1.MDAMB231.SRR2046032\_2M.trimmed.fastq ILLUMINACLIP:TruSeq3-SE.fa:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:33

Automatically using 1 threads

Using Long Clipping Sequence: 'AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGTA'

Using Long Clipping Sequence: 'AGATCGGAAGAGCACACGTCTGAACTCCAGTCAC'

ILLUMINACLIP: Using 0 prefix pairs, 2 forward/reverse sequences, 0 forward only sequences, 0 reverse only sequences

Input Reads: 2000000 Surviving: 1686012 (84.30%) Dropped: 313988 (15.70%)

TrimmomaticSE: Completed successfully

TrimmomaticSE: Started with arguments:

-phred33 /home/jupyterAbiVatsa/bigdata/ProjectWork\_sequences/set1/vehi\_NAh\_rep2.MDAMB231.SRR2046033\_2M.fastq /home/jupyterAbiVatsa/trimmed\_reads/vehi\_NAh\_rep2.MDAMB231.SRR2046033\_2M.trimmed.fastq ILLUMINACLIP:TruSeq3-SE.fa:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:33

Automatically using 1 threads

Using Long Clipping Sequence: 'AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGTA'

Using Long Clipping Sequence: 'AGATCGGAAGAGCACACGTCTGAACTCCAGTCAC'

ILLUMINACLIP: Using 0 prefix pairs, 2 forward/reverse sequences, 0 forward only sequences, 0 reverse only sequences

Input Reads: 2000000 Surviving: 1983067 (99.15%) Dropped: 16933 (0.85%)

TrimmomaticSE: Completed successfully

### **Quality control for trimmed reads**

In [9]:

%%bash

mkdir -p "**${**ProjectFolder**}**/FastQC\_trimmed"

files=**$(**find "**${**ProjectFolder**}**/trimmed\_reads" | grep ".fastq$"**)**

fastqc **\**

$files **\**

-t 1 **\**

-o "**${**ProjectFolder**}**/FastQC\_trimmed" **\**

--quiet

echo "All done!"

All done!

### **Explore the .html FastQC reports for the trimmed data**

### **Checking library information**

Create a subset of one trimmed FASTQ file

In [10]:

%%bash

*# Make folder for tests*

mkdir -p "**${**ProjectFolder**}**/tests\_folder"

*# Call the tool to subset reads*

python3 "/home/jupyterAbiVatsaPractical2/subsetFastq.py" **\**

"**${**ProjectFolder**}**/trimmed\_reads/dexa\_2h\_rep1.MDAMB231.SRR2046034\_2M.trimmed.fastq" **\**

"**${**ProjectFolder**}**/tests\_folder/test.fastq" **\**

--number 1000

counting records....

sampling 1000 out of 1983162 records

5.042452406812958 % done

10.084904813625917 % done

15.127357220438874 % done

20.169809627251833 % done

25.212262034064793 % done

30.25471444087775 % done

35.29716684769071 % done

40.33961925450367 % done

45.38207166131662 % done

50.424524068129585 % done

55.46697647494254 % done

60.5094288817555 % done

65.55188128856845 % done

70.59433369538142 % done

75.63678610219438 % done

80.67923850900733 % done

85.72169091582029 % done

90.76414332263325 % done

95.80659572944622 % done

done!

### **Perform preliminary alignment**

In [11]:

%%bash

*# Running HISAT2 only with standard parameters*

hisat2 **\**

-x "/home/jupyterAbiVatsa/bigdata/HISAT\_indices/hg38\_tran/genome\_tran" **\**

-U "**${**ProjectFolder**}**/tests\_folder/test.fastq" **\**

-S "**${**ProjectFolder**}**/tests\_folder/test\_strand.sam"

1000 reads; of these:

1000 (100.00%) were unpaired; of these:

19 (1.90%) aligned 0 times

825 (82.50%) aligned exactly 1 time

156 (15.60%) aligned >1 times

98.10% overall alignment rate

### **Inferring library type information from the test alignment**

In [12]:

%%bash

infer\_experiment.py **\**

-r "/home/jupyterAbiVatsa/Practical2/hg38\_RefSeq.bed" **\**

-i "**${**ProjectFolder**}**/tests\_folder/test\_strand.sam"

This is SingleEnd Data

Fraction of reads failed to determine: 0.0593

Fraction of reads explained by "++,--": 0.4747

Fraction of reads explained by "+-,-+": 0.4660

Reading reference gene model /home/jupyterAbiVatsa/Practical2/hg38\_RefSeq.bed ... Done

Loading SAM/BAM file ... Finished

Total 809 usable reads were sampled

Now that we have result from sampled trimmed read, and we can see that more than 98% of the reads were aligned successfully we can now go for full trimmed raw data reads to align.

### **Alignment**

\*\*NOTE:\*\* It may be necessary to use also additional settings for HISAT2. See Practical 2 notebook for guidance. Additional help: <http://daehwankimlab.github.io/hisat2/manual/>.

In [3]:

%%bash

mkdir -p "**${**ProjectFolder**}**/HISAT\_aligned"

**for** file in **$(**ls "**${**ProjectFolder**}**/trimmed\_reads" | grep ".trimmed.fastq$"**)**; **do**

filename=`echo **${**file**}** | sed 's/.trimmed.fastq//'`

outFile="**${**ProjectFolder**}**/HISAT\_aligned/**${**filename**}**.sam"

hisat2 **\**

-x "/home/jupyterAbiVatsa/bigdata/HISAT\_indices/hg38\_tran/genome\_tran" **\**

-U "**${**ProjectFolder**}**/trimmed\_reads/**${**file**}**" **\**

-S "**${**outFile**}**" **\**

--summary-file "**${**ProjectFolder**}**/HISAT\_aligned/**${**filename**}**.summary.txt" **\**

-k 2 **\**

--mp 6,2

**done**

1983162 reads; of these:

1983162 (100.00%) were unpaired; of these:

39421 (1.99%) aligned 0 times

1646105 (83.00%) aligned exactly 1 time

297636 (15.01%) aligned >1 times

98.01% overall alignment rate

1688880 reads; of these:

1688880 (100.00%) were unpaired; of these:

33892 (2.01%) aligned 0 times

1406915 (83.30%) aligned exactly 1 time

248073 (14.69%) aligned >1 times

97.99% overall alignment rate

1975678 reads; of these:

1975678 (100.00%) were unpaired; of these:

51430 (2.60%) aligned 0 times

1625191 (82.26%) aligned exactly 1 time

299057 (15.14%) aligned >1 times

97.40% overall alignment rate

1684798 reads; of these:

1684798 (100.00%) were unpaired; of these:

31727 (1.88%) aligned 0 times

1408895 (83.62%) aligned exactly 1 time

244176 (14.49%) aligned >1 times

98.12% overall alignment rate

1686012 reads; of these:

1686012 (100.00%) were unpaired; of these:

43122 (2.56%) aligned 0 times

1394441 (82.71%) aligned exactly 1 time

248449 (14.74%) aligned >1 times

97.44% overall alignment rate

1983067 reads; of these:

1983067 (100.00%) were unpaired; of these:

35224 (1.78%) aligned 0 times

1650116 (83.21%) aligned exactly 1 time

297727 (15.01%) aligned >1 times

98.22% overall alignment rate

Now we have successfully aligned more than 97% in each of the trimmed raw reads that we had to reference genome.

### **Converting SAM files to BAM files, and generating BAM index files**

In [4]:

%%bash

**for** file in **$(**ls "**${**ProjectFolder**}**/HISAT\_aligned" | grep ".sam$"**)**; **do**

filename=`echo **${**file**}** | sed 's/.sam//'`

outFile="**${**ProjectFolder**}**/HISAT\_aligned/**${**filename**}**.sorted.bam"

samtools sort -O BAM "**${**ProjectFolder**}**/HISAT\_aligned/**${**file**}**" -o "**${**outFile**}**"

samtools index "**${**outFile**}**"

**done**

Now we have successfully created aligned files in binary compressed BAM format

### **Alignment quality control with RSeQC**

#### **Checking chromosome format for compatibility**

In [5]:

%%bash

*# View the BAM file with samtools view.*

samtools view "**${**ProjectFolder**}**/HISAT\_aligned/dexa\_2h\_rep1.MDAMB231.SRR2046034\_2M.sorted.bam" | head -2

SRR2046034.21818526 0 chr1 14520 1 50M \* 0 0 CCGCCCCAGCTGTGTGGCCTCAAGCCAGCCTTCCGCTCCTTGAAGCTGGT CCCFFFFFGHGGHIGIJIJJJJIIIJEIJJJJJJJJJJIJJIJIFJJJJG AS:i:0 ZS:i:0 XN:i:0 XM:i:0 XO:i:0 XG:i:0 NM:i:0 MD:Z:50 YT:Z:UU NH:i:2

SRR2046034.31867245 256 chr1 14582 1 50M \* 0 0 CTGGTTCCGTCACCCCCTCCCAAGGAAGTAGGTCTGAGCAGCTTGTCCTG CCCFFFFFHGHDHGIJJIJJJIIJJIJH?GHHFGIIIJJGHJJGIHIJJJ AS:i:0 ZS:i:0 XN:i:0 XM:i:0 XO:i:0 XG:i:0 NM:i:0 MD:Z:50 YT:Z:UU NH:i:2

#### **Basic BAM statistics and read distribution statistics**

In [6]:

%%bash

**for** file in **$(**ls "**${**ProjectFolder**}**/HISAT\_aligned" | grep ".bam$"**)**; **do**

bam\_stat.py -i "**${**ProjectFolder**}**/HISAT\_aligned/**${**file**}**"

**done**

#==================================================

#All numbers are READ count

#==================================================

Total records: 2280798

QC failed: 0

Optical/PCR duplicate: 0

Non primary hits 297636

Unmapped reads: 39421

mapq < mapq\_cut (non-unique): 297636

mapq >= mapq\_cut (unique): 1646105

Read-1: 0

Read-2: 0

Reads map to '+': 825826

Reads map to '-': 820279

Non-splice reads: 1371265

Splice reads: 274840

Reads mapped in proper pairs: 0

Proper-paired reads map to different chrom:0

#==================================================

#All numbers are READ count

#==================================================

Total records: 1936953

QC failed: 0

Optical/PCR duplicate: 0

Non primary hits 248073

Unmapped reads: 33892

mapq < mapq\_cut (non-unique): 248073

mapq >= mapq\_cut (unique): 1406915

Read-1: 0

Read-2: 0

Reads map to '+': 704632

Reads map to '-': 702283

Non-splice reads: 1176835

Splice reads: 230080

Reads mapped in proper pairs: 0

Proper-paired reads map to different chrom:0

#==================================================

#All numbers are READ count

#==================================================

Total records: 2274735

QC failed: 0

Optical/PCR duplicate: 0

Non primary hits 299057

Unmapped reads: 51430

mapq < mapq\_cut (non-unique): 299057

mapq >= mapq\_cut (unique): 1625191

Read-1: 0

Read-2: 0

Reads map to '+': 816031

Reads map to '-': 809160

Non-splice reads: 1359278

Splice reads: 265913

Reads mapped in proper pairs: 0

Proper-paired reads map to different chrom:0

#==================================================

#All numbers are READ count

#==================================================

Total records: 1928974

QC failed: 0

Optical/PCR duplicate: 0

Non primary hits 244176

Unmapped reads: 31727

mapq < mapq\_cut (non-unique): 244176

mapq >= mapq\_cut (unique): 1408895

Read-1: 0

Read-2: 0

Reads map to '+': 706086

Reads map to '-': 702809

Non-splice reads: 1174417

Splice reads: 234478

Reads mapped in proper pairs: 0

Proper-paired reads map to different chrom:0

#==================================================

#All numbers are READ count

#==================================================

Total records: 1934461

QC failed: 0

Optical/PCR duplicate: 0

Non primary hits 248449

Unmapped reads: 43122

mapq < mapq\_cut (non-unique): 248449

mapq >= mapq\_cut (unique): 1394441

Read-1: 0

Read-2: 0

Reads map to '+': 698711

Reads map to '-': 695730

Non-splice reads: 1165890

Splice reads: 228551

Reads mapped in proper pairs: 0

Proper-paired reads map to different chrom:0

#==================================================

#All numbers are READ count

#==================================================

Total records: 2280794

QC failed: 0

Optical/PCR duplicate: 0

Non primary hits 297727

Unmapped reads: 35224

mapq < mapq\_cut (non-unique): 297727

mapq >= mapq\_cut (unique): 1650116

Read-1: 0

Read-2: 0

Reads map to '+': 826329

Reads map to '-': 823787

Non-splice reads: 1375090

Splice reads: 275026

Reads mapped in proper pairs: 0

Proper-paired reads map to different chrom:0

Load BAM file ... Done

Load BAM file ... Done

Load BAM file ... Done

Load BAM file ... Done

Load BAM file ... Done

Load BAM file ... Done

In [7]:

%%bash

**for** file in **$(**ls "**${**ProjectFolder**}**/HISAT\_aligned" | grep ".bam$"**)**; **do**

read\_distribution.py -i "**${**ProjectFolder**}**/HISAT\_aligned/**${**file**}**" **\**

-r "/home/jupyterAbiVatsa/Practical2/hg38\_RefSeq.bed"

**done**

Total Reads 1943741

Total Tags 2236288

Total Assigned Tags 2080624

=====================================================================

Group Total\_bases Tag\_count Tags/Kb

CDS\_Exons 39184887 1368763 34.93

5'UTR\_Exons 37214570 91215 2.45

3'UTR\_Exons 58010731 468947 8.08

Introns 1539911688 92195 0.06

TSS\_up\_1kb 24518972 8444 0.34

TSS\_up\_5kb 110212503 27669 0.25

TSS\_up\_10kb 201827137 43002 0.21

TES\_down\_1kb 27333233 4212 0.15

TES\_down\_5kb 118264548 10810 0.09

TES\_down\_10kb 210002921 16502 0.08

=====================================================================

Total Reads 1654988

Total Tags 1899835

Total Assigned Tags 1770069

=====================================================================

Group Total\_bases Tag\_count Tags/Kb

CDS\_Exons 39184887 1161304 29.64

5'UTR\_Exons 37214570 76890 2.07

3'UTR\_Exons 58010731 395216 6.81

Introns 1539911688 87766 0.06

TSS\_up\_1kb 24518972 7004 0.29

TSS\_up\_5kb 110212503 23274 0.21

TSS\_up\_10kb 201827137 35337 0.18

TES\_down\_1kb 27333233 3585 0.13

TES\_down\_5kb 118264548 9064 0.08

TES\_down\_10kb 210002921 13556 0.06

=====================================================================

Total Reads 1924248

Total Tags 2207768

Total Assigned Tags 2052431

=====================================================================

Group Total\_bases Tag\_count Tags/Kb

CDS\_Exons 39184887 1328963 33.92

5'UTR\_Exons 37214570 86854 2.33

3'UTR\_Exons 58010731 471142 8.12

Introns 1539911688 108046 0.07

TSS\_up\_1kb 24518972 8779 0.36

TSS\_up\_5kb 110212503 27838 0.25

TSS\_up\_10kb 201827137 42520 0.21

TES\_down\_1kb 27333233 4003 0.15

TES\_down\_5kb 118264548 9911 0.08

TES\_down\_10kb 210002921 14906 0.07

=====================================================================

Total Reads 1653071

Total Tags 1902868

Total Assigned Tags 1774428

=====================================================================

Group Total\_bases Tag\_count Tags/Kb

CDS\_Exons 39184887 1170282 29.87

5'UTR\_Exons 37214570 75276 2.02

3'UTR\_Exons 58010731 396299 6.83

Introns 1539911688 85343 0.06

TSS\_up\_1kb 24518972 6960 0.28

TSS\_up\_5kb 110212503 22337 0.20

TSS\_up\_10kb 201827137 34401 0.17

TES\_down\_1kb 27333233 3474 0.13

TES\_down\_5kb 118264548 8749 0.07

TES\_down\_10kb 210002921 12827 0.06

=====================================================================

Total Reads 1642890

Total Tags 1886268

Total Assigned Tags 1755466

=====================================================================

Group Total\_bases Tag\_count Tags/Kb

CDS\_Exons 39184887 1152687 29.42

5'UTR\_Exons 37214570 75237 2.02

3'UTR\_Exons 58010731 391963 6.76

Introns 1539911688 86410 0.06

TSS\_up\_1kb 24518972 7135 0.29

TSS\_up\_5kb 110212503 23342 0.21

TSS\_up\_10kb 201827137 35475 0.18

TES\_down\_1kb 27333233 3578 0.13

TES\_down\_5kb 118264548 9203 0.08

TES\_down\_10kb 210002921 13694 0.07

=====================================================================

Total Reads 1947843

Total Tags 2241217

Total Assigned Tags 2085674

=====================================================================

Group Total\_bases Tag\_count Tags/Kb

CDS\_Exons 39184887 1371176 34.99

5'UTR\_Exons 37214570 91322 2.45

3'UTR\_Exons 58010731 456150 7.86

Introns 1539911688 107683 0.07

TSS\_up\_1kb 24518972 8388 0.34

TSS\_up\_5kb 110212503 27829 0.25

TSS\_up\_10kb 201827137 42863 0.21

TES\_down\_1kb 27333233 4384 0.16

TES\_down\_5kb 118264548 11042 0.09

TES\_down\_10kb 210002921 16480 0.08

=====================================================================

processing /home/jupyterAbiVatsa/Practical2/hg38\_RefSeq.bed ... Done

processing /home/jupyterAbiVatsa/HISAT\_aligned/dexa\_2h\_rep1.MDAMB231.SRR2046034\_2M.sorted.bam ... Finished

processing /home/jupyterAbiVatsa/Practical2/hg38\_RefSeq.bed ... Done

processing /home/jupyterAbiVatsa/HISAT\_aligned/dexa\_2h\_rep2.MDAMB231.SRR2046035\_2M.sorted.bam ... Finished

processing /home/jupyterAbiVatsa/Practical2/hg38\_RefSeq.bed ... Done

processing /home/jupyterAbiVatsa/HISAT\_aligned/dexa\_4h\_rep1.MDAMB231.SRR2046036\_2M.sorted.bam ... Finished

processing /home/jupyterAbiVatsa/Practical2/hg38\_RefSeq.bed ... Done

processing /home/jupyterAbiVatsa/HISAT\_aligned/dexa\_4h\_rep2.MDAMB231.SRR2046037\_2M.sorted.bam ... Finished

processing /home/jupyterAbiVatsa/Practical2/hg38\_RefSeq.bed ... Done

processing /home/jupyterAbiVatsa/HISAT\_aligned/vehi\_NAh\_rep1.MDAMB231.SRR2046032\_2M.sorted.bam ... Finished

processing /home/jupyterAbiVatsa/Practical2/hg38\_RefSeq.bed ... Done

processing /home/jupyterAbiVatsa/HISAT\_aligned/vehi\_NAh\_rep2.MDAMB231.SRR2046033\_2M.sorted.bam ... Finished

### **Read counting and annotation**

\*\*NOTES:\*\*

* Set your HISAT2 output folder in the HISATfolder variable in R.
* Insert correct parameters for featureCounts before running it. See Practical 3 notebook for guidance.

In [3]:

%%R

library(Rsubread)

*# Set input folder yourself!*

HISATfolder <- "/home/jupyterAbiVatsa/HISAT\_aligned"

bamFiles <- list.files(HISATfolder, pattern = ".bam$", full.names=**TRUE**)

featureCounts\_object <- featureCounts(bamFiles,

annot.inbuilt="hg38",

GTF.featureType="exon",

GTF.attrType="gene\_id",

strandSpecific = 0,

isPairedEnd = **FALSE**,

countMultiMappingReads = **FALSE**,

countChimericFragments = **FALSE**,

minMQS=0,

useMetaFeatures=**TRUE**)

NCBI RefSeq annotation for hg38 (build 38.2) is used.

========== \_\_\_\_\_ \_ \_ \_\_\_\_ \_\_\_\_\_ \_\_\_\_\_\_ \_\_\_\_\_

===== / \_\_\_\_| | | | \_ \| \_\_ \| \_\_\_\_| /\ | \_\_ \

===== | (\_\_\_ | | | | |\_) | |\_\_) | |\_\_ / \ | | | |

==== \\_\_\_ \| | | | \_ <| \_ /| \_\_| / /\ \ | | | |

==== \_\_\_\_) | |\_\_| | |\_) | | \ \| |\_\_\_\_ / \_\_\_\_ \| |\_\_| |

========== |\_\_\_\_\_/ \\_\_\_\_/|\_\_\_\_/|\_| \\_\\_\_\_\_\_\_/\_/ \\_\\_\_\_\_\_/

Rsubread 1.34.7

//========================== featureCounts setting ===========================\\

|| ||

|| Input files : 6 BAM files ||

|| o dexa\_2h\_rep1.MDAMB231.SRR2046034\_2M.sorted ... ||

|| o dexa\_2h\_rep2.MDAMB231.SRR2046035\_2M.sorted ... ||

|| o dexa\_4h\_rep1.MDAMB231.SRR2046036\_2M.sorted ... ||

|| o dexa\_4h\_rep2.MDAMB231.SRR2046037\_2M.sorted ... ||

|| o vehi\_NAh\_rep1.MDAMB231.SRR2046032\_2M.sorte ... ||

|| o vehi\_NAh\_rep2.MDAMB231.SRR2046033\_2M.sorte ... ||

|| ||

|| Annotation : inbuilt (hg38) ||

|| Dir for temp files : . ||

|| Threads : 1 ||

|| Level : meta-feature level ||

|| Paired-end : no ||

|| Multimapping reads : not counted ||

|| Multi-overlapping reads : not counted ||

|| Min overlapping bases : 1 ||

|| ||

\\============================================================================//

//================================= Running ==================================\\

|| ||

|| Load annotation file hg38\_RefSeq\_exon.txt ... ||

|| Features : 261752 ||

|| Meta-features : 28395 ||

|| Chromosomes/contigs : 55 ||

|| ||

|| Process BAM file dexa\_2h\_rep1.MDAMB231.SRR2046034\_2M.sorted.bam... ||

|| Single-end reads are included. ||

|| Total alignments : 2280798 ||

|| Successfully assigned alignments : 1524203 (66.8%) ||

|| Running time : 0.06 minutes ||

|| ||

|| Process BAM file dexa\_2h\_rep2.MDAMB231.SRR2046035\_2M.sorted.bam... ||

|| Single-end reads are included. ||

|| Total alignments : 1936953 ||

|| Successfully assigned alignments : 1293368 (66.8%) ||

|| Running time : 0.05 minutes ||

|| ||

|| Process BAM file dexa\_4h\_rep1.MDAMB231.SRR2046036\_2M.sorted.bam... ||

|| Single-end reads are included. ||

|| Total alignments : 2274735 ||

|| Successfully assigned alignments : 1489497 (65.5%) ||

|| Running time : 0.06 minutes ||

|| ||

|| Process BAM file dexa\_4h\_rep2.MDAMB231.SRR2046037\_2M.sorted.bam... ||

|| Single-end reads are included. ||

|| Total alignments : 1928974 ||

|| Successfully assigned alignments : 1298086 (67.3%) ||

|| Running time : 0.05 minutes ||

|| ||

|| Process BAM file vehi\_NAh\_rep1.MDAMB231.SRR2046032\_2M.sorted.bam... ||

|| Single-end reads are included. ||

|| Total alignments : 1934461 ||

|| Successfully assigned alignments : 1281261 (66.2%) ||

|| Running time : 0.06 minutes ||

|| ||

|| Process BAM file vehi\_NAh\_rep2.MDAMB231.SRR2046033\_2M.sorted.bam... ||

|| Single-end reads are included. ||

|| Total alignments : 2280794 ||

|| Successfully assigned alignments : 1511182 (66.3%) ||

|| Running time : 0.07 minutes ||

|| ||

|| ||

\\============================================================================//

In [5]:

%%R

featureCounts\_object$stat

Status dexa.2h.rep1.MDAMB231.SRR2046034.2M.sorted.bam

1 Assigned 1524203

2 Unassigned\_Unmapped 39421

3 Unassigned\_Read\_Type 0

4 Unassigned\_Singleton 0

5 Unassigned\_MappingQuality 0

6 Unassigned\_Chimera 0

7 Unassigned\_FragmentLength 0

8 Unassigned\_Duplicate 0

9 Unassigned\_MultiMapping 595272

10 Unassigned\_Secondary 0

11 Unassigned\_NonSplit 0

12 Unassigned\_NoFeatures 84161

13 Unassigned\_Overlapping\_Length 0

14 Unassigned\_Ambiguity 37741

dexa.2h.rep2.MDAMB231.SRR2046035.2M.sorted.bam

1 1293368

2 33892

3 0

4 0

5 0

6 0

7 0

8 0

9 496146

10 0

11 0

12 81751

13 0

14 31796

dexa.4h.rep1.MDAMB231.SRR2046036.2M.sorted.bam

1 1489497

2 51430

3 0

4 0

5 0

6 0

7 0

8 0

9 598114

10 0

11 0

12 97840

13 0

14 37854

dexa.4h.rep2.MDAMB231.SRR2046037.2M.sorted.bam

1 1298086

2 31727

3 0

4 0

5 0

6 0

7 0

8 0

9 488352

10 0

11 0

12 79127

13 0

14 31682

vehi.NAh.rep1.MDAMB231.SRR2046032.2M.sorted.bam

1 1281261

2 43122

3 0

4 0

5 0

6 0

7 0

8 0

9 496898

10 0

11 0

12 80654

13 0

14 32526

vehi.NAh.rep2.MDAMB231.SRR2046033.2M.sorted.bam

1 1511182

2 35224

3 0

4 0

5 0

6 0

7 0

8 0

9 595454

10 0

11 0

12 100329

13 0

14 38605

### **Exploring the results of featureCounts**

In [4]:

%%R

counts <- featureCounts\_object$counts

\*\*NOTE:\*\* If you want to do *e.g.* small scale heatmaps and display the gene symbols using information stored in rowData, you have to add it here.

In [5]:

%%R

*# Add gene symbols to row annotations*

*# Load package that contains annotatioon information for*

library(org.Hs.eg.db)

*# Get vector of gene IDs. We will use the row names of the count matrix here.*

geneIDs <- row.names(counts)

*# Map entrez gene IDs to gene symbols*

rowData\_init<- select(org.Hs.eg.db, keys=geneIDs, columns="SYMBOL", keytype="ENTREZID")

*# Since there might be cases where multiple HGHC symbols map to single EntrezID lets concatenate rows*

rowData\_init <- aggregate(SYMBOL ~ ENTREZID, data = rowData\_init, paste, collapse = ";")

*# Make temporary annotation data frame with first column as Entrez IDs*

rowData\_counts <- data.frame(ENTREZID=geneIDs)

*# Merge the two data frames*

rowData <- merge(x = rowData\_counts, y = rowData\_init, all.x = **TRUE**, by = 'ENTREZID', incomparables = **NA**)

*# Setting row names for the data frame helps in ordering, like in the next step (also rows can be called upon by their names, not just by their indices)*

rownames(rowData) <- rowData$ENTREZID

*# Just for safety, lets order the rows as they are in counts matrix*

rowData <- rowData[geneIDs,]

*# Now we can check the first few rows of rowData*

head(rowData)

ENTREZID SYMBOL

100287102 100287102 DDX11L1

653635 653635 WASH7P

102466751 102466751 MIR6859-1

100302278 100302278 MIR1302-2

645520 645520 FAM138A

79501 79501 OR4F5

\*\*NOTE:\*\* If you want to use *e.g.* advanced heatmap annotation you can add more annotation levels to colData for treatment times and/or treatments here.

In [6]:

%%R

colData <- data.frame(Filename=featureCounts\_object$targets,

Group=gsub("\\.rep.\*","",featureCounts\_object$targets), *# creates groups signatures by removing everything starting from "\_rep" from the file names*

Treatment=gsub("\\..\*","",featureCounts\_object$targets),

Time=rep(c("2h","4h","NAh"), each=2))

colData

Filename Group Treatment Time

1 dexa.2h.rep1.MDAMB231.SRR2046034.2M.sorted.bam dexa.2h dexa 2h

2 dexa.2h.rep2.MDAMB231.SRR2046035.2M.sorted.bam dexa.2h dexa 2h

3 dexa.4h.rep1.MDAMB231.SRR2046036.2M.sorted.bam dexa.4h dexa 4h

4 dexa.4h.rep2.MDAMB231.SRR2046037.2M.sorted.bam dexa.4h dexa 4h

5 vehi.NAh.rep1.MDAMB231.SRR2046032.2M.sorted.bam vehi.NAh vehi NAh

6 vehi.NAh.rep2.MDAMB231.SRR2046033.2M.sorted.bam vehi.NAh vehi NAh

\*\*NOTE:\*\* Generate SummarizedExperiment here for QC analysis.

\*\*NOTE:\*\* It probably makes sense to save the se object to file after generating it below, to provide a convenient starting point for (highly likely necessary) tuning rounds of the Report figures for the sample QC analysis. See *e.g.* Practical 3 notebook for examples.

In [7]:

%%R

library(SummarizedExperiment)

se <- SummarizedExperiment(assays=list(counts=counts),

rowData=rowData,

colData=colData)

*# Save se object as RDS*

*#ProjectFolder <- Sys.getenv("ProjectFolder")*

*#file\_rds <- paste(ProjectFolder, "/SummarizedExperiment.RDS", sep="")*

*#save(se, file=file\_rds)*

### **Sample-level quality control**

In [9]:

%%R

library(DESeq2)

*# Calculate sample-wise read count sums*

as.matrix(colSums(assay(se)))

[,1]

dexa.2h.rep1.MDAMB231.SRR2046034.2M.sorted.bam 1524203

dexa.2h.rep2.MDAMB231.SRR2046035.2M.sorted.bam 1293368

dexa.4h.rep1.MDAMB231.SRR2046036.2M.sorted.bam 1489497

dexa.4h.rep2.MDAMB231.SRR2046037.2M.sorted.bam 1298086

vehi.NAh.rep1.MDAMB231.SRR2046032.2M.sorted.bam 1281261

vehi.NAh.rep2.MDAMB231.SRR2046033.2M.sorted.bam 1511182

In [10]:

%%R

dds <- DESeqDataSet(se, design = ~1)

vst <- varianceStabilizingTransformation(dds)

\*\*NOTE:\*\* Set proper size parameters for the plot.

In [11]:

%%R

pcaData.vst <- plotPCA(vst, intgroup=c("Group"), returnData=**TRUE**)

pcaData.vst

PC1 PC2 group

dexa.2h.rep1.MDAMB231.SRR2046034.2M.sorted.bam -3.635618 -3.6473414 dexa.2h

dexa.2h.rep2.MDAMB231.SRR2046035.2M.sorted.bam -3.529626 -3.9698290 dexa.2h

dexa.4h.rep1.MDAMB231.SRR2046036.2M.sorted.bam -8.520951 3.5973938 dexa.4h

dexa.4h.rep2.MDAMB231.SRR2046037.2M.sorted.bam -8.402840 2.1895350 dexa.4h

vehi.NAh.rep1.MDAMB231.SRR2046032.2M.sorted.bam 12.159865 -0.2344454 vehi.NAh

vehi.NAh.rep2.MDAMB231.SRR2046033.2M.sorted.bam 11.929170 2.0646871 vehi.NAh

Group

dexa.2h.rep1.MDAMB231.SRR2046034.2M.sorted.bam dexa.2h

dexa.2h.rep2.MDAMB231.SRR2046035.2M.sorted.bam dexa.2h

dexa.4h.rep1.MDAMB231.SRR2046036.2M.sorted.bam dexa.4h

dexa.4h.rep2.MDAMB231.SRR2046037.2M.sorted.bam dexa.4h

vehi.NAh.rep1.MDAMB231.SRR2046032.2M.sorted.bam vehi.NAh

vehi.NAh.rep2.MDAMB231.SRR2046033.2M.sorted.bam vehi.NAh

name

dexa.2h.rep1.MDAMB231.SRR2046034.2M.sorted.bam dexa.2h.rep1.MDAMB231.SRR2046034.2M.sorted.bam

dexa.2h.rep2.MDAMB231.SRR2046035.2M.sorted.bam dexa.2h.rep2.MDAMB231.SRR2046035.2M.sorted.bam

dexa.4h.rep1.MDAMB231.SRR2046036.2M.sorted.bam dexa.4h.rep1.MDAMB231.SRR2046036.2M.sorted.bam

dexa.4h.rep2.MDAMB231.SRR2046037.2M.sorted.bam dexa.4h.rep2.MDAMB231.SRR2046037.2M.sorted.bam

vehi.NAh.rep1.MDAMB231.SRR2046032.2M.sorted.bam vehi.NAh.rep1.MDAMB231.SRR2046032.2M.sorted.bam

vehi.NAh.rep2.MDAMB231.SRR2046033.2M.sorted.bam vehi.NAh.rep2.MDAMB231.SRR2046033.2M.sorted.bam

In [12]:

%%R

*# Show the attributes related to the data.frame*

attributes(pcaData.vst)

$names

[1] "PC1" "PC2" "group" "Group" "name"

$class

[1] "data.frame"

$row.names

[1] "dexa.2h.rep1.MDAMB231.SRR2046034.2M.sorted.bam"

[2] "dexa.2h.rep2.MDAMB231.SRR2046035.2M.sorted.bam"

[3] "dexa.4h.rep1.MDAMB231.SRR2046036.2M.sorted.bam"

[4] "dexa.4h.rep2.MDAMB231.SRR2046037.2M.sorted.bam"

[5] "vehi.NAh.rep1.MDAMB231.SRR2046032.2M.sorted.bam"

[6] "vehi.NAh.rep2.MDAMB231.SRR2046033.2M.sorted.bam"

$percentVar

[1] 0.84444410 0.09402842

In [13]:

%%R -w 700 -h 600

library(ggplot2)

*# Change the variance proportions into percentages, and round*

percentVar.vst <- round(100 \* attr(pcaData.vst, "percentVar"))

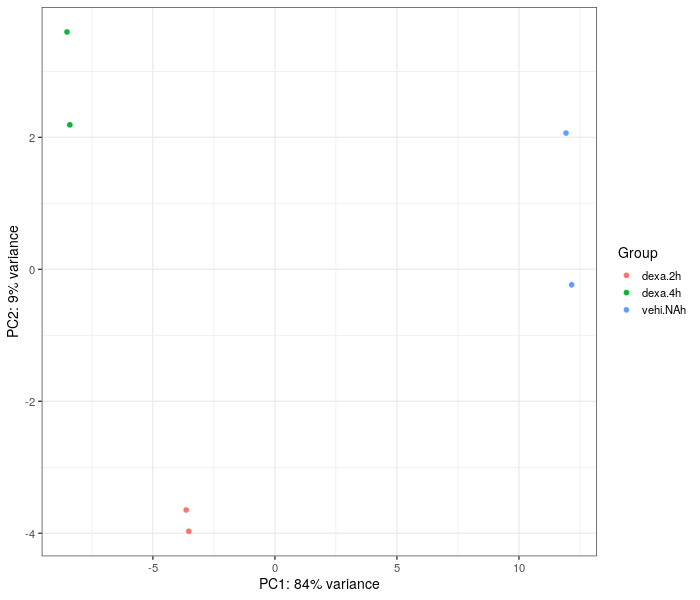
ggplot(pcaData.vst, aes(PC1, PC2, color=Group)) +

geom\_point(size=2) +

xlab(paste0("PC1: ",percentVar.vst[1],"% variance")) +

ylab(paste0("PC2: ",percentVar.vst[2],"% variance")) +

theme\_bw(base\_size=14)



We can see that the principal component plot has been able to place the samples into three separate regions distinctively

\*\*NOTE:\*\* Set proper size parameters for the plot.

\*\*NOTE:\*\* In contrast to Practical 4 where this step aimed for up to 10K of genes, with the project data a smaller set of genes will likely work even better, because the difference between the groups are far less drastic.

In [14]:

%%R

*# Set this yourself.*

percentage <- 0.95

vst.mat <- assay(vst)

*# Get named vector of standard deviations and print it's length (= the total number of genes)*

sds <- apply(vst.mat, MARGIN=1, FUN="sd")

*# Print the total number of genes using 'cat()'. Note that 'cat()' doesn't automatically feed the*

*# line, so the newline character ("\n") must be given at the end to prevent the next line printing*

*# on the same line.*

cat("total number of genes:", length(sds), "\n")

*# Which rows have SD above the selected percentage of all SDs*

sel <- (sds>quantile(sds,percentage))

*# Subset the rows*

vst.subset <- vst.mat[sel, ]

*# Print the number of rows*

cat("number of subset genes:", nrow(vst.subset), "\n")

total number of genes: 28395

number of subset genes: 1420

In [15]:

%%R -w 600 -h 700

library(ComplexHeatmap)

library(circlize)

*# Lets center the counts by substracting row means from the row values*

scaled.vst <- vst.subset-rowMeans(vst.subset)

colnames(scaled.vst) <- gsub("\\.M..\*", "", colnames(scaled.vst))

*# Set the color scale*

col\_fun = colorRamp2(c(min(scaled.vst),0, max(scaled.vst)), c("blue", "white","red"))

dist.name <- "euclidean"

Heatmap(scaled.vst,

col = col\_fun,

show\_row\_names = F,

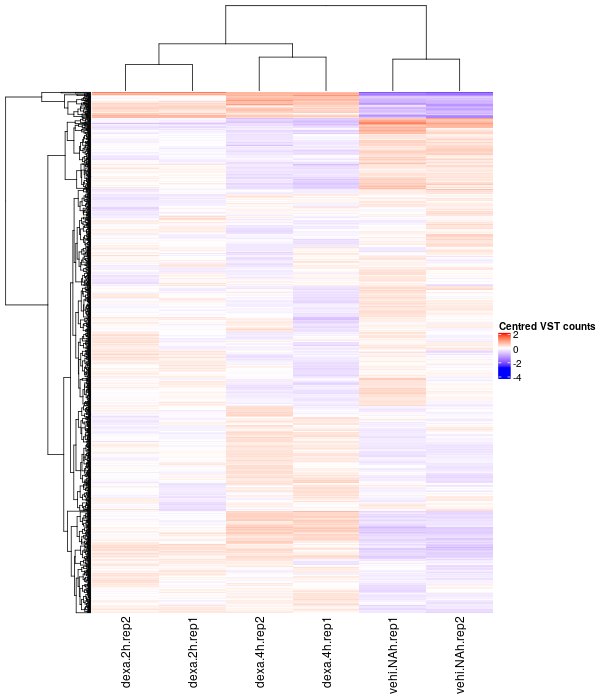
clustering\_distance\_rows = dist.name,

clustering\_distance\_columns = dist.name,

row\_dend\_width = unit(3, "cm"),

column\_dend\_height = unit(3, "cm"),

name="Centred VST counts")



### **Creating DGEList object**

\*\*NOTE:\*\* Even though in Practical 3 we added gene and sample annotation to a DGEList, ***do not*** add anything extra to the object here!

\*\*NOTE:\*\* Below, set the filepath for the output file.

In [16]:

%%R

library(edgeR)

dge <- DGEList(counts = counts, group = gsub("\\.rep..\*", "", colnames(counts)))

*#saveRDS(dge, "/home/jupyter*AbiVatsa*/projectTeam4.RDS")*

# **Differential expression analysis**

In [17]:

%%R

library(limma)

library(Homo.sapiens)

x <- readRDS(paste0("/home/jupyterAbiVatsa", "/projectTeam4.RDS"))

x

An object of class "DGEList"

$counts

dexa.2h.rep1.MDAMB231.SRR2046034.2M.sorted.bam

100287102 0

653635 3

102466751 0

100302278 0

645520 0

dexa.2h.rep2.MDAMB231.SRR2046035.2M.sorted.bam

100287102 0

653635 5

102466751 0

100302278 0

645520 0

dexa.4h.rep1.MDAMB231.SRR2046036.2M.sorted.bam

100287102 0

653635 3

102466751 0

100302278 0

645520 0

dexa.4h.rep2.MDAMB231.SRR2046037.2M.sorted.bam

100287102 0

653635 2

102466751 0

100302278 0

645520 0

vehi.NAh.rep1.MDAMB231.SRR2046032.2M.sorted.bam

100287102 0

653635 0

102466751 0

100302278 0

645520 0

vehi.NAh.rep2.MDAMB231.SRR2046033.2M.sorted.bam

100287102 0

653635 2

102466751 0

100302278 0

645520 0

28390 more rows ...

$samples

group lib.size norm.factors

dexa.2h.rep1.MDAMB231.SRR2046034.2M.sorted.bam dexa.2h 1524203 1

dexa.2h.rep2.MDAMB231.SRR2046035.2M.sorted.bam dexa.2h 1293368 1

dexa.4h.rep1.MDAMB231.SRR2046036.2M.sorted.bam dexa.4h 1489497 1

dexa.4h.rep2.MDAMB231.SRR2046037.2M.sorted.bam dexa.4h 1298086 1

vehi.NAh.rep1.MDAMB231.SRR2046032.2M.sorted.bam vehi.NAh 1281261 1

vehi.NAh.rep2.MDAMB231.SRR2046033.2M.sorted.bam vehi.NAh 1511182 1

In [20]:

%%R

dim(x)

[1] 28395 6

### **Organising gene annotations**

In [18]:

%%R

geneid <- rownames(x)

genes <- select(Homo.sapiens, keys=geneid, columns=c("SYMBOL", "TXCHROM"), keytype="ENTREZID")

head(genes)

ENTREZID SYMBOL TXCHROM

1 100287102 DDX11L1 chr1

2 653635 WASH7P chr1

3 102466751 MIR6859-1 <NA>

4 100302278 MIR1302-2 <NA>

5 645520 FAM138A <NA>

6 79501 OR4F5 chr1

In [19]:

%%R

*# Remove duplicates*

genes <- genes[!duplicated(genes$ENTREZID),]

In [20]:

%%R

x$genes <- genes

x

An object of class "DGEList"

$counts

dexa.2h.rep1.MDAMB231.SRR2046034.2M.sorted.bam

100287102 0

653635 3

102466751 0

100302278 0

645520 0

dexa.2h.rep2.MDAMB231.SRR2046035.2M.sorted.bam

100287102 0

653635 5

102466751 0

100302278 0

645520 0

dexa.4h.rep1.MDAMB231.SRR2046036.2M.sorted.bam

100287102 0

653635 3

102466751 0

100302278 0

645520 0

dexa.4h.rep2.MDAMB231.SRR2046037.2M.sorted.bam

100287102 0

653635 2

102466751 0

100302278 0

645520 0

vehi.NAh.rep1.MDAMB231.SRR2046032.2M.sorted.bam

100287102 0

653635 0

102466751 0

100302278 0

645520 0

vehi.NAh.rep2.MDAMB231.SRR2046033.2M.sorted.bam

100287102 0

653635 2

102466751 0

100302278 0

645520 0

28390 more rows ...

$samples

group lib.size norm.factors

dexa.2h.rep1.MDAMB231.SRR2046034.2M.sorted.bam dexa.2h 1524203 1

dexa.2h.rep2.MDAMB231.SRR2046035.2M.sorted.bam dexa.2h 1293368 1

dexa.4h.rep1.MDAMB231.SRR2046036.2M.sorted.bam dexa.4h 1489497 1

dexa.4h.rep2.MDAMB231.SRR2046037.2M.sorted.bam dexa.4h 1298086 1

vehi.NAh.rep1.MDAMB231.SRR2046032.2M.sorted.bam vehi.NAh 1281261 1

vehi.NAh.rep2.MDAMB231.SRR2046033.2M.sorted.bam vehi.NAh 1511182 1

$genes

ENTREZID SYMBOL TXCHROM

1 100287102 DDX11L1 chr1

2 653635 WASH7P chr1

3 102466751 MIR6859-1 <NA>

4 100302278 MIR1302-2 <NA>

5 645520 FAM138A <NA>

28390 more rows ...

## **Data pre-processing**

### **Transformations from the raw-scale**

In [21]:

%%R

cpm <- cpm(x)

lcpm <- cpm(x, log=**TRUE**)

L <- mean(x$samples$lib.size) \* 1e-6

M <- median(x$samples$lib.size) \* 1e-6

c(L, M)

summary(lcpm)

dexa.2h.rep1.MDAMB231.SRR2046034.2M.sorted.bam

Min. : 0.515

1st Qu.: 0.515

Median : 0.515

Mean : 2.479

3rd Qu.: 4.684

Max. :12.727

dexa.2h.rep2.MDAMB231.SRR2046035.2M.sorted.bam

Min. : 0.515

1st Qu.: 0.515

Median : 0.515

Mean : 2.485

3rd Qu.: 4.667

Max. :12.736

dexa.4h.rep1.MDAMB231.SRR2046036.2M.sorted.bam

Min. : 0.515

1st Qu.: 0.515

Median : 0.515

Mean : 2.468

3rd Qu.: 4.600

Max. :12.933

dexa.4h.rep2.MDAMB231.SRR2046037.2M.sorted.bam

Min. : 0.515

1st Qu.: 0.515

Median : 0.515

Mean : 2.467

3rd Qu.: 4.617

Max. :12.743

vehi.NAh.rep1.MDAMB231.SRR2046032.2M.sorted.bam

Min. : 0.515

1st Qu.: 0.515

Median : 0.515

Mean : 2.494

3rd Qu.: 4.723

Max. :12.621

vehi.NAh.rep2.MDAMB231.SRR2046033.2M.sorted.bam

Min. : 0.515

1st Qu.: 0.515

Median : 0.515

Mean : 2.494

3rd Qu.: 4.696

Max. :12.686

### **Removing genes that are lowly expressed**

In [22]:

%%R

*# Number of genes with zero counts across all samples*

table(rowSums(x$counts==0)==6)

FALSE TRUE

16083 12312

In [23]:

%%R

keep.exprs <- filterByExpr(x, group=x$samples$group)

x <- x[keep.exprs,, keep.lib.sizes=**FALSE**]

dim(x)

[1] 10247 6

In [24]:

%%R

samplenames <- substring(colnames(x), 1, 13)

lcpm.cutoff <- log2(10/M + 2/L)

library(RColorBrewer)

nsamples <- ncol(x)

col <- brewer.pal(nsamples, "Paired")

par(mfrow=c(1,2))

plot(density(lcpm[,1]), col=col[1], lwd=2, ylim=c(0,0.26), las=2, main="", xlab="")

title(main="A. Raw data", xlab="Log-cpm")

abline(v=lcpm.cutoff, lty=3)

for (i in 2:nsamples){

den <- density(lcpm[,i])

lines(den$x, den$y, col=col[i], lwd=2)

}

legend("topright", samplenames, text.col=col, bty="n")

lcpm <- cpm(x, log=**TRUE**)

plot(density(lcpm[,1]), col=col[1], lwd=2, ylim=c(0,0.26), las=2, main="", xlab="")

title(main="B. Filtered data", xlab="Log-cpm")

abline(v=lcpm.cutoff, lty=3)

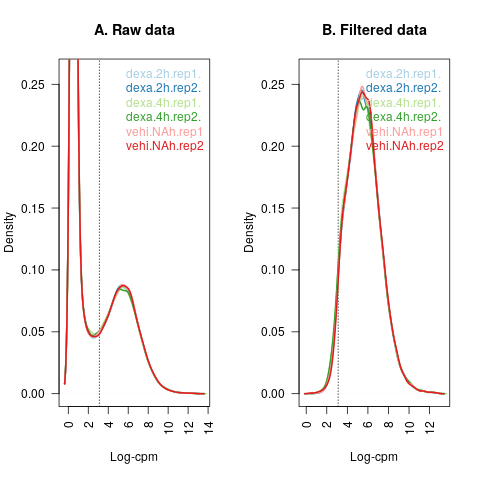
for (i in 2:nsamples){

den <- density(lcpm[,i])

lines(den$x, den$y, col=col[i], lwd=2)

}

legend("topright", samplenames, text.col=col, bty="n")



### **Normalising gene expression distributions**

In [25]:

%%R

samplenames <- substring(colnames(x),1,8)

par(mfrow=c(1,2))

lcpm <- cpm(x, log=**TRUE**)

boxplot(lcpm, las=2, col=col, main="", names=samplenames)

title(main="A. Unnormalised data",ylab="Log-cpm")

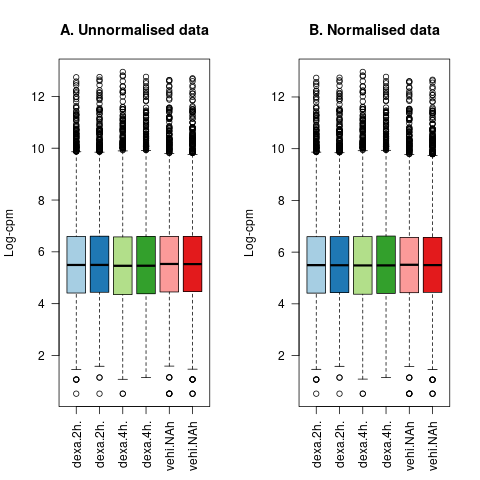
x <- calcNormFactors(x, method = "TMM")

x$samples$norm.factors

lcpm <- cpm(x, log=**TRUE**)

boxplot(lcpm, las=2, col=col, main="", names=samplenames)

title(main="B. Normalised data",ylab="Log-cpm")



### **Unsupervised clustering of samples**

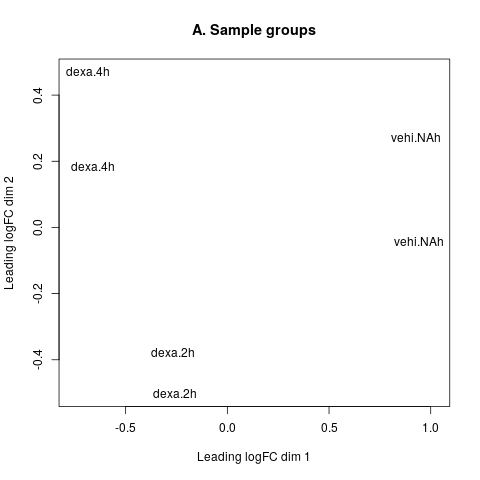
In [26]:

%%R

lcpm <- cpm(x, log=**TRUE**)

plotMDS(lcpm, labels=x$samples$group)

title(main="A. Sample groups")



### **Creating design matrix and contrasts**

In [27]:

%%R

design <- model.matrix(~0+x$samples$group)

colnames(design) <- c("dexa.2h","dexa.4h","vehi.NAh")

design

dexa.2h dexa.4h vehi.NAh

1 1 0 0

2 1 0 0

3 0 1 0

4 0 1 0

5 0 0 1

6 0 0 1

attr(,"assign")

[1] 1 1 1

attr(,"contrasts")

attr(,"contrasts")$`x$samples$group`

[1] "contr.treatment"

In [28]:

%%R

contr.matrix <- makeContrasts(

dexa2hvsdexa4h = dexa.2h-dexa.4h,

dexa2hvsvehiNAh = dexa.2h - vehi.NAh,

dexa4hvsvehiNAh = dexa.4h - vehi.NAh,

levels = colnames(design))

contr.matrix

Contrasts

Levels dexa2hvsdexa4h dexa2hvsvehiNAh dexa4hvsvehiNAh

dexa.2h 1 1 0

dexa.4h -1 0 1

vehi.NAh 0 -1 -1

### **Removing heteroscedascity from count data and fitting linear models for comparisons of interest**

In [29]:

%%R

par(mfrow=c(1,2))

v <- voom(x, design, plot=**TRUE**)

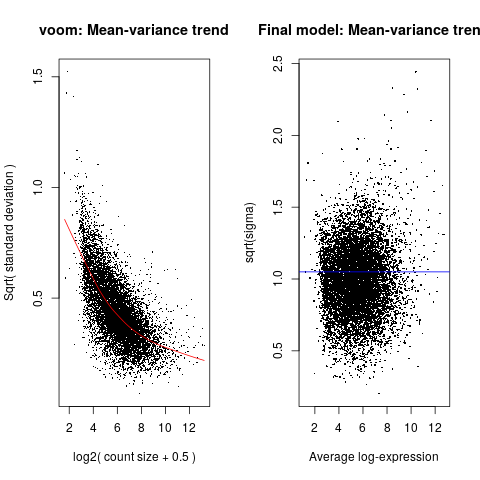
v

vfit <- lmFit(v, design)

vfit <- contrasts.fit(vfit, contrasts=contr.matrix)

efit <- eBayes(vfit)

plotSA(efit, main="Final model: Mean-variance trend")



### **Examining the number of DE genes**

In [30]:

%%R

dt <- decideTests(efit)

summary(dt)

dexa2hvsdexa4h dexa2hvsvehiNAh dexa4hvsvehiNAh

Down 125 160 430

NotSig 9976 9854 9357

Up 146 233 460

In [31]:

%%R

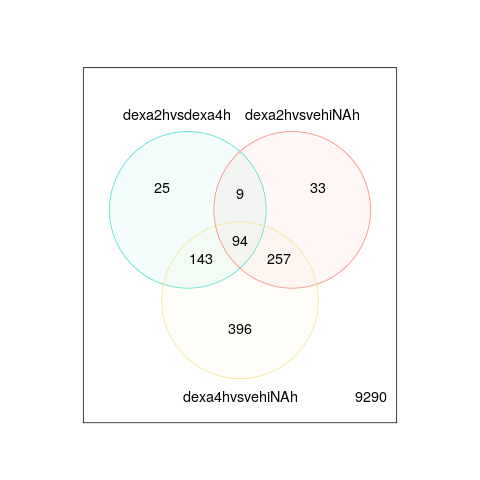
de.common <- which(dt[,1]!=0 & dt[,2]!=0 & dt[,3]!=0)

length(de.common)

head(efit$genes$SYMBOL[de.common], n=20)

vennDiagram(dt[,1:3], circle.col=c("turquoise", "salmon", "khaki"), cex=1.2)

*#write.fit(efit, dt, file="results.txt")*



### **Examining individual DE genes from top to bottom**

In [32]:

%%R

dexa2h.vs.dexa4h <- topTable(efit, coef=1, n=**Inf**)

head(dexa2h.vs.dexa4h)

*#write.table(dexa2h.vs.dexa4h, "dexa2h\_vs\_dexa4h.txt", row.names=FALSE)*

ENTREZID SYMBOL TXCHROM logFC AveExpr t P.Value

2289 2289 FKBP5 chr6 -1.4956442 9.089879 -20.914904 7.366190e-21

23268 23268 DNMBP chr10 1.1555592 7.672515 9.406968 6.037072e-11

128434 128434 VSTM2L chr20 -0.9315925 7.880523 -8.954277 2.002004e-10

2180 2180 ACSL1 chr4 -1.0539323 7.112129 -8.854097 2.619628e-10

4502 4502 MT2A chr16 -0.6582336 9.765853 -8.013044 2.631917e-09

10458 10458 BAIAP2 chr17 -0.8996397 7.300353 -7.967011 2.993646e-09

adj.P.Val B

2289 7.548135e-17 37.11553

23268 3.093094e-07 14.89399

128434 6.710833e-07 13.81566

2180 6.710833e-07 13.51276

4502 5.112648e-06 11.19955

10458 5.112648e-06 11.18456

In [33]:

%%R

dexa2h.vs.vehiNAh <- topTable(efit, coef=2, n=**Inf**)

head(dexa2h.vs.vehiNAh)

*#write.table(dexa2h.vs.dexa4h, "dexa2h\_vs\_vehiNAh.txt", row.names=FALSE)*

ENTREZID SYMBOL TXCHROM logFC AveExpr t P.Value

2289 2289 FKBP5 chr6 3.608075 9.089879 28.79927 2.731329e-25

1831 1831 TSC22D3 chrX 5.103374 8.276275 28.61514 3.361483e-25

10628 10628 TXNIP chr1 2.154054 9.854216 25.41540 1.530394e-23

54541 54541 DDIT4 chr10 2.933701 8.510682 25.57361 1.254446e-23

54206 54206 ERRFI1 chr1 2.183652 10.542454 23.29931 2.447544e-22

6574 6574 SLC20A1 chr2 1.967338 9.704938 20.93196 7.180544e-21

adj.P.Val B

2289 1.722256e-21 46.99274

1831 1.722256e-21 45.60715

10628 3.920486e-20 43.70135

54541 3.920486e-20 43.59584

54206 5.015997e-19 40.96807

6574 1.226317e-17 37.60254

In [34]:

%%R

dexa4h.vs.vehiNAh <- topTable(efit, coef=3, n=**Inf**)

head(dexa4h.vs.vehiNAh)

*#write.table(dexa2h.vs.dexa4h, "dexa4h\_vs\_vehiNAh.txt", row.names=FALSE)*

ENTREZID SYMBOL TXCHROM logFC AveExpr t P.Value

2289 2289 FKBP5 chr6 5.103720 9.089879 42.22011 9.898201e-31

1831 1831 TSC22D3 chrX 5.112578 8.276275 28.65396 3.217208e-25

4502 4502 MT2A chr16 2.303075 9.765853 24.09557 8.407090e-23

54541 54541 DDIT4 chr10 2.744438 8.510682 23.72622 1.374644e-22

4493 4493 MT1E chr16 3.123666 8.065572 22.81402 4.770071e-22

10628 10628 TXNIP chr1 1.843248 9.854216 21.35405 3.839375e-21

adj.P.Val B

2289 1.014269e-26 58.58826

1831 1.648336e-21 46.10259

4502 2.871582e-19 42.03288

54541 3.521494e-19 41.40897

4493 9.775783e-19 40.12544

10628 6.167484e-18 38.21966

In [35]:

%%R

plotMD(efit, column=1, status=dt[,1], main=colnames(efit)[1],

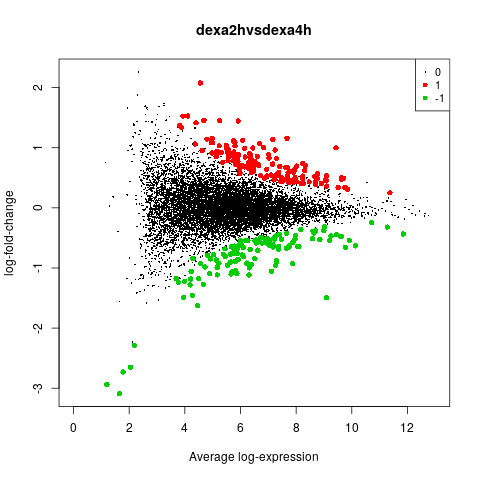
xlim=c(0,13))

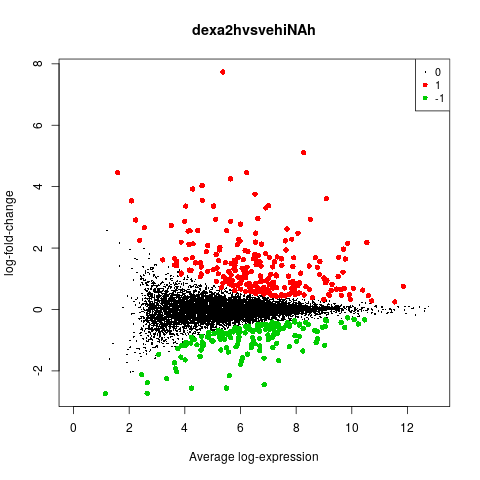
plotMD(efit, column=2, status=dt[,2], main=colnames(efit)[2],

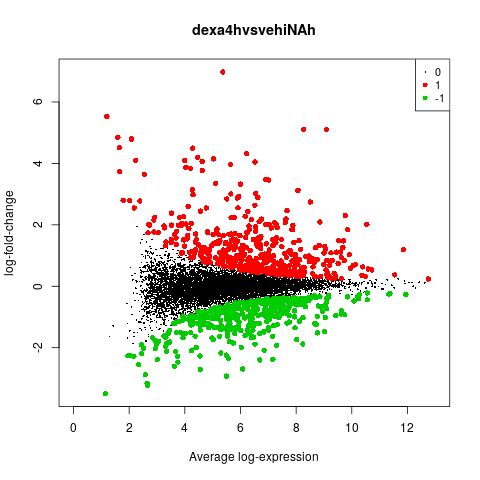
xlim=c(0,13))

plotMD(efit, column=3, status=dt[,3], main=colnames(efit)[3],

xlim=c(0,13))







In [42]:

%%R

dex <- dexa4h.vs.vehiNAh$ENTREZID[1:100]

i <- x[x$genes$ENTREZID %in% dex,]

In [62]:

%%R -w 600 -h 800

*# Heatmap of top 100 DE genes in Dex4h vs. vehiNAh*

lcpm.values <- cpm(i$counts, log=**TRUE**)

colnames(lcpm.values) <- gsub("\\.M..\*", "", colnames(lcpm.values))

rownames(lcpm.values) <- i$genes$SYMBOL

dist.name <- "euclidean"

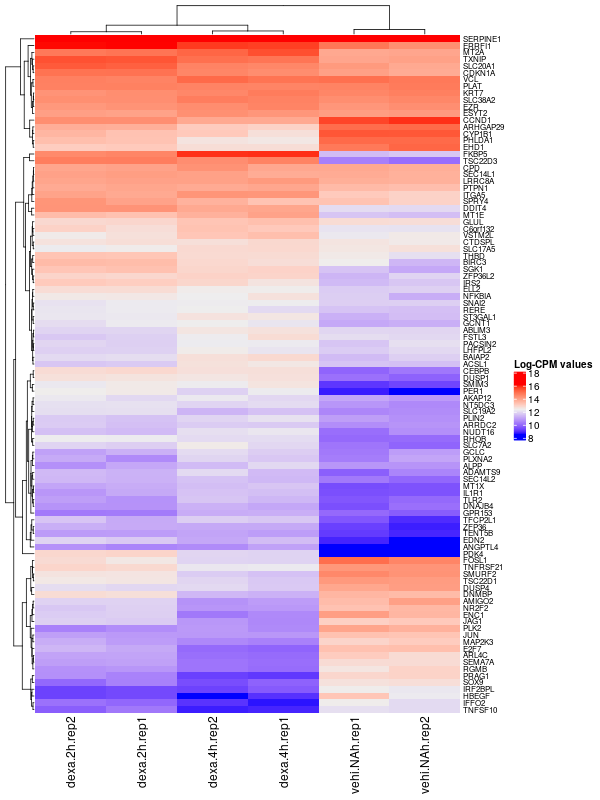
Heatmap(lcpm.values,

name="Log-CPM values",

row\_names\_gp = gpar(fontsize = 8),

clustering\_distance\_rows = dist.name,

clustering\_distance\_columns = dist.name)



In [ ]:

%%R

*# Selecting genes with logFC >= 1 or logFC <= -1 from each comparison and saving them to a file*

dexa4h.vs.vehiNAhlogFCg1 = dexa4h.vs.vehiNAh[dexa4h.vs.vehiNAh$logFC > 1,]

write.csv(dexa4h.vs.vehiNAhlogFCg1$SYMBOL, "~/course\_data/SEQDA23Teams/Team4/dexa4h.vs.vehiNAhEnriched.csv", row.names=**FALSE**)

dexa4h.vs.vehiNAhlogFClminus1 = dexa4h.vs.vehiNAh[dexa4h.vs.vehiNAh$logFC < -1,]

write.csv(dexa4h.vs.vehiNAhlogFClminus1$SYMBOL, "~/course\_data/SEQDA23Teams/Team4/dexa4h.vs.vehiNAhRepressed.csv", row.names=**FALSE**)

dexa2h.vs.vehiNAhlogFClminus1 = dexa2h.vs.vehiNAh[dexa2h.vs.vehiNAh$logFC < -1,]

write.csv(dexa2h.vs.vehiNAhlogFClminus1$SYMBOL, "~/course\_data/SEQDA23Teams/Team4/dexa2h.vs.vehiNAhRepressed.csv", row.names=**FALSE**)

dexa2h.vs.vehiNAhlogFCg1 = dexa2h.vs.vehiNAh[dexa2h.vs.vehiNAh$logFC > 1,]

write.csv(dexa2h.vs.vehiNAhlogFCg1$SYMBOL, "~/course\_data/SEQDA23Teams/Team4/dexa2h.vs.vehiNAhEnriched.csv", row.names=**FALSE**)

dexa2h.vs.dexa4hlogFCg1 = dexa2h.vs.dexa4h[dexa2h.vs.dexa4h$logFC > 1,]

write.csv(dexa2h.vs.dexa4hlogFCg1$SYMBOL, "~/course\_data/SEQDA23Teams/Team4/dexa2h.vs.dexa4hEnriched.csv", row.names=**FALSE**)

dexa2h.vs.dexa4hlogFClminus1 = dexa2h.vs.dexa4h[dexa2h.vs.dexa4h$logFC < -1,]

write.csv(dexa2h.vs.dexa4hlogFClminus1$SYMBOL, "~/course\_data/SEQDA23Teams/Team4/dexa2h.vs.dexa4hRepressed.csv", row.names=**FALSE**)

**SUPPLEMENTARY MATERIALS**

Chart, bar chart

Description automatically generated

**Supplementary Figure 1. Per base quality scores prior to read trimming.** The distribution of quality scores at each position in the read across all reads in the sample Dex\_2h\_rep1 are plotted in the graph. In the plot, the red line represents the median value, yellow box represents the inter-quartile range (25-75%), the upper and lower whiskers represent the 10% and 90% points, and the blue line indicates the mean quality. The quality scores are shown on the y-axis and base pair position on the x-axis.

Chart, line chart

Description automatically generated

**Supplementary Figure 2. Quality score distribution of sequences prior to read trimming.** The plot represents the distribution of average quality scores in the sample Dex\_2h\_rep1. The average quality score is shown on the x-axis and the number of sequences with the given average on the y-axis.

Chart

Description automatically generated

**Supplementary Figure 3. Per base quality scores after read trimming.** The distribution of quality scores at each position in the read across all reads in the sample Dex\_2h\_rep1 after conducting read trimming are shown in the plot. The quality scores on average rise in the lower quartile after discarding low quality bases from the reads.

Chart, line chart

Description automatically generated

**Supplementary Figure 4.** **Quality score distribution of sequences after read trimming.** The plot depicts the distribution of average quality scores in the sample Dex\_2h\_rep1 after discarding low quality bases. Distributions of quality scores in other samples were almost identical to the provided example, meaning that all samples had only high-quality sequences left after read trimming was conducted.

Chart, box and whisker chart

Description automatically generated with medium confidence

**Supplementary Figure 5. A clustered heatmap of centred gene expression counts.** Sample-level quality assessment included plotting of read counts of a subset of genes into a heatmap to inspect and visualize differences in gene expression across the samples and genes. Red colour indicates high gene expression and blue lower gene expression. Rows and columns were clustered by Euclidean distance.