recount_brain example with data from SRP027383

true

Abstract

This is an example on how to use recount_brain applied to the SRP027383 study. We show how to download data from recount2, add the sample metadata from recount_brain, explore the sample metadata and the gene expression data, and perform a gene expression analysis.

Introduction

This document is an example of how you can use `recount_brain`. We will use the data from the SRA study SRP027383 which is described in “RNA-seq of 272 gliomas revealed a novel, recurrent PTPRZ1-MET fusion transcript in secondary glioblastomas” (Bao, Chen, Yang, Zhang, et al., 2014). As you can see in Figure @ref(fig:runselector) a lot of the metadata for these samples is missing from the SRA Run Selector which makes it a great case for using `recount_brain`. We will show how to add the `recount_brain` metadata and perform a gene differential expression analysis using this information.

Sample metadata

Just like any study in `recount2` (Collado-Torres, Nellore, Kammers, Ellis, et al., 2017), we first need to download the gene count data using `recount::download_study()`. Since we will be using many functions from the `recount` package, lets load it first[1].

```
## Load the package
library('recount')
```

Download gene data

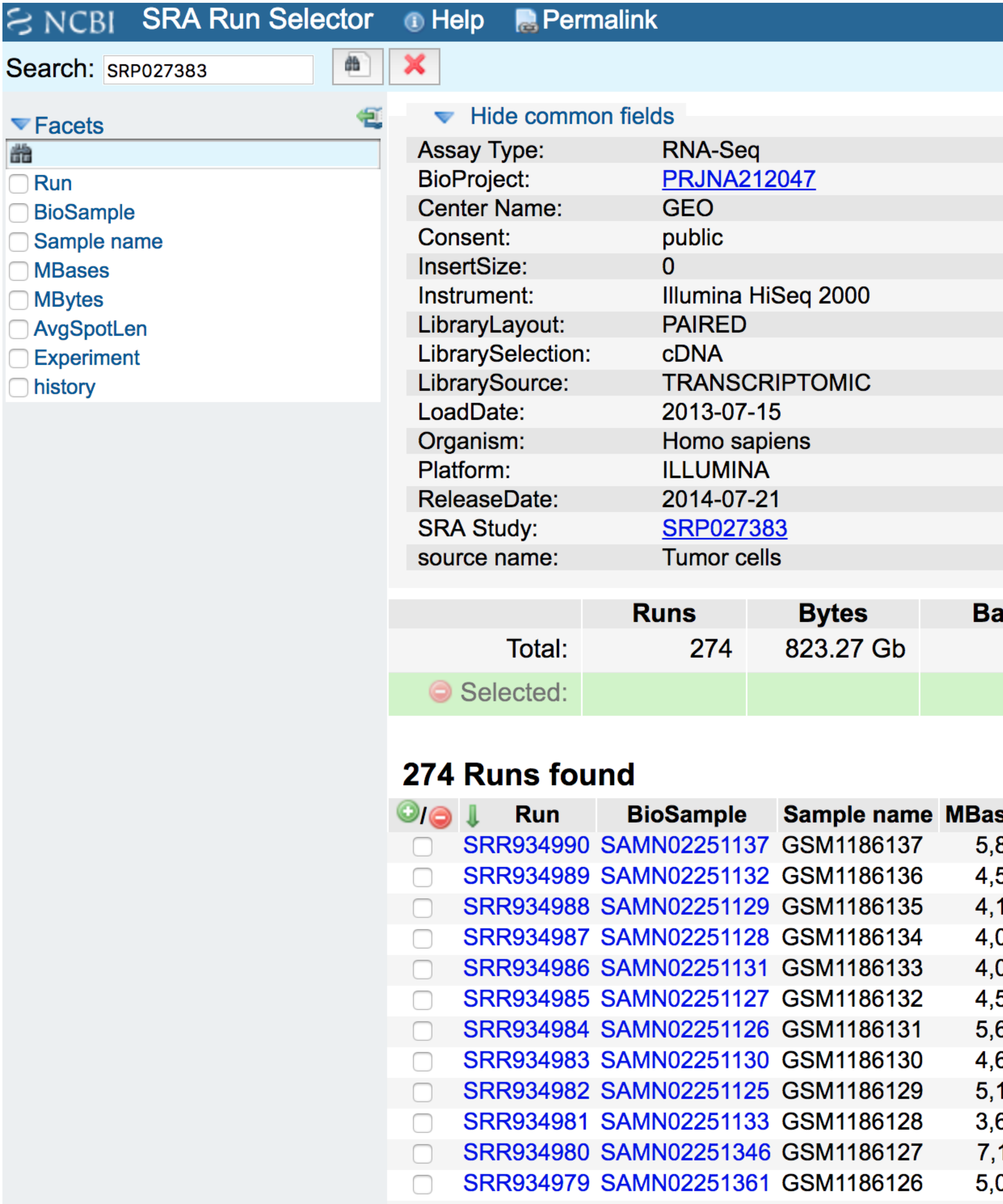
Having loaded the package, we next download the gene-level data.

```
if(!file.exists(file.path('SRP027383', 'rse_gene.Rdata'))) {
    download_study('SRP027383')
}
load(file.path('SRP027383', 'rse_gene.Rdata'), verbose = TRUE)
```

```
## Loading objects:
##   rse_gene
```

---

[1] If you are a first time `recount` user, we recommend first reading the package vignette at bioconductor.org/packages/recount.



Search: SRP027383

Assay Type:	RNA-Seq
BioProject:	PRJNA212047
Center Name:	GEO
Consent:	public
InsertSize:	0
Instrument:	Illumina HiSeq 2000
LibraryLayout:	PAIRED
LibrarySelection:	cDNA
LibrarySource:	TRANSCRIPTOMIC
LoadDate:	2013-07-15
Organism:	Homo sapiens
Platform:	ILLUMINA
ReleaseDate:	2014-07-21
SRA Study:	SRP027383
source name:	Tumor cells

	Runs	Bytes	Ba
Total:	274	823.27 Gb	
Selected:			

274 Runs found

	Run	BioSample	Sample name	MBas
☐	SRR934990	SAMN02251137	GSM1186137	5,8
☐	SRR934989	SAMN02251132	GSM1186136	4,5
☐	SRR934988	SAMN02251129	GSM1186135	4,1
☐	SRR934987	SAMN02251128	GSM1186134	4,0
☐	SRR934986	SAMN02251131	GSM1186133	4,0
☐	SRR934985	SAMN02251127	GSM1186132	4,5
☐	SRR934984	SAMN02251126	GSM1186131	5,6
☐	SRR934983	SAMN02251130	GSM1186130	4,6
☐	SRR934982	SAMN02251125	GSM1186129	5,1
☐	SRR934981	SAMN02251133	GSM1186128	3,6
☐	SRR934980	SAMN02251346	GSM1186127	7,1
☐	SRR934979	SAMN02251361	GSM1186126	5,0

Figure 1: SRA Run Selector information for study SRP027383. Screenshot from 2018-02-26.

Sample metadata included in `recount`

We can next explore the sample metadata that is included by default using `SummarizedExperiment::colData()`. These variables are explained in more detail in the supplementary material of the `recount2` paper (Collado-Torres, Nellore, Kammers, Ellis, et al., 2017) and in the `recount workflow` paper (Collado-Torres, Nellore, and Jaffe, 2017).

```r
colData(rse_gene)
```

```
## DataFrame with 270 rows and 21 columns
##               project      sample  experiment         run read_count_as_reported_by_sra reads_downloa
##           <character> <character> <character> <character>                     <integer>        <inte
## SRR934717   SRP027383   SRS457680   SRX322602   SRR934717                      56887576         5688
## SRR934718   SRP027383   SRS457681   SRX322603   SRR934718                      39683692         3968
## SRR934719   SRP027383   SRS457682   SRX322604   SRR934719                      39392540         3939
## SRR934720   SRP027383   SRS457683   SRX322605   SRR934720                      60287388         6028
## SRR934721   SRP027383   SRS457684   SRX322606   SRR934721                      31089346         3108
## ...               ...         ...         ...         ...                           ...
## SRR934986   SRP027383   SRS457949   SRX322871   SRR934986                      42563170         4256
## SRR934987   SRP027383   SRS457950   SRX322872   SRR934987                      42481802         4248
## SRR934988   SRP027383   SRS457951   SRX322873   SRR934988                      43121132         4312
## SRR934989   SRP027383   SRS457952   SRX322874   SRR934989                      47384314         4738
## SRR934990   SRP027383   SRS457953   SRX322875   SRR934990                      61093682         6109
##           proportion_of_reads_reported_by_sra_downloaded paired_end sra_misreported_paired_end mappe
##                                                <numeric>  <logical>                  <logical>
## SRR934717                                              1       TRUE                      FALSE
## SRR934718                                              1       TRUE                      FALSE
## SRR934719                                              1       TRUE                      FALSE
## SRR934720                                              1       TRUE                      FALSE
## SRR934721                                              1       TRUE                      FALSE
## ...                                                  ...        ...                        ...
## SRR934986                                              1       TRUE                      FALSE
## SRR934987                                              1       TRUE                      FALSE
## SRR934988                                              1       TRUE                      FALSE
## SRR934989                                              1       TRUE                      FALSE
## SRR934990                                              1       TRUE                      FALSE
##                  auc sharq_beta_tissue sharq_beta_cell_type biosample_submission_date biosample_publi
##            <numeric>       <character>          <character>               <character>
## SRR934717 5628071616    umbilical cord                  esc   2013-07-15T11:26:36.860    2014-07-20T
## SRR934718 3950872208    umbilical cord                  esc   2013-07-15T11:28:33.710    2014-07-20T
## SRR934719 3958083805    umbilical cord                  esc   2013-07-15T11:26:47.540    2014-07-20T
## SRR934720 6047049537    umbilical cord                  esc   2013-07-15T11:26:44.253    2014-07-20T
## SRR934721 3072882301    umbilical cord                  esc   2013-07-15T11:28:18.330    2014-07-20T
## ...              ...               ...                  ...                       ...
## SRR934986 4259218453    umbilical cord                  esc   2013-07-15T11:22:27.600    2014-07-20T
## SRR934987 4245759225    umbilical cord                  esc   2013-07-15T11:22:07.083    2014-07-20T
## SRR934988 4309934199    umbilical cord                  esc   2013-07-15T11:22:10.270    2014-07-20T
## SRR934989 4739386115    umbilical cord                  esc   2013-07-15T11:22:37.680    2014-07-20T
## SRR934990 6110940825    umbilical cord                  esc   2013-07-15T11:23:19.253    2014-07-20T
##             biosample_update_date avg_read_length geo_accession  bigwig_file       title
##                       <character>       <integer>   <character>  <character> <character>
## SRR934717 2014-07-20T01:22:14.790             202    GSM1185864 SRR934717.bw    CGGA_171
## SRR934718 2014-07-20T01:22:14.977             200    GSM1185865 SRR934718.bw    CGGA_235
## SRR934719 2014-07-20T01:22:15.377             202    GSM1185866 SRR934719.bw    CGGA_236
```

```
## SRR934720 2014-07-20T01:22:15.650             202    GSM1185867 SRR934720.bw    CGGA_241
## SRR934721 2014-07-20T01:22:16.003             200    GSM1185868 SRR934721.bw    CGGA_243
## ...                             ...             ...           ...          ...         ...
## SRR934986 2014-07-20T01:15:29.503             202    GSM1186133 SRR934986.bw   CGGA_J030
## SRR934987 2014-07-20T01:18:22.877             202    GSM1186134 SRR934987.bw   CGGA_J042
## SRR934988 2014-07-20T01:18:23.733             202    GSM1186135 SRR934988.bw   CGGA_J100
## SRR934989 2014-07-20T01:18:24.270             202    GSM1186136 SRR934989.bw   CGGA_J130
## SRR934990 2014-07-20T01:18:25.100             202    GSM1186137 SRR934990.bw   CGGA_J023
##                                    characteristics
##                                    <CharacterList>
## SRR934717           history: oligodendroastrocytomas
## SRR934718           history: oligodendroastrocytomas
## SRR934719                history: oligodendrogliomas
## SRR934720           history: oligodendroastrocytomas
## SRR934721           history: oligodendroastrocytomas
## ...                                              ...
## SRR934986           history: oligodendroastrocytomas
## SRR934987 history: recurrent oligodendroastrocytomas
## SRR934988           history: recurrent Glioblastomas
## SRR934989            history: recurrent astrocytomas
## SRR934990     history: anaplastic oligodendrogliomas
```

Note how the `characteristics` column matches the information from the SRA Run Selector in Figure @ref(fig:runselector). Still not very useful.

```
colData(rse_gene)$characteristics
```

```
## CharacterList of length 270
## [[1]] history: oligodendroastrocytomas
## [[2]] history: oligodendroastrocytomas
## [[3]] history: oligodendrogliomas
## [[4]] history: oligodendroastrocytomas
## [[5]] history: oligodendroastrocytomas
## [[6]] history: recurrent astrocytomas
## [[7]] history: oligodendroastrocytomas
## [[8]] history: astrocytomas
## [[9]] history: oligodendroastrocytomas
## [[10]] history: astrocytomas
## ...
## <260 more elements>
```

Add `recount_brain` sample metadata

So lets add the available sample metadata from `recount_brain` using the `recount::add_metadata()` function.

```
rse_gene <- add_metadata(rse = rse_gene, source = 'recount_brain_v1')
```

```
## 2020-11-13 16:27:22 downloading the recount_brain metadata to /tmp/RtmpK9pZcs/recount_brain_v1.Rdata
```

```
## Loading objects:
##   recount_brain
```

```
## 2020-11-13 16:27:23 found 270 out of 270 samples in the recount_brain metadata
```

Explore `recount_brain` metadata

We can now explore the available metadata from `recount_brain` for the SRP027383 study.

```
## Find which new columns have observations
new_non_NA <- sapply(22:ncol(colData(rse_gene)),
    function(i) any(!is.na(colData(rse_gene)[, i])) )
## Display the observations
colData(rse_gene)[, (22:ncol(colData(rse_gene)))[new_non_NA]]
```

```
## DataFrame with 270 rows and 33 columns
##           assay_type_s avgspotlen_l bioproject_s  biosample_s center_name_s   consent_s disease_statu
##            <character>    <integer>  <character>  <character>   <character> <character>    <characte
## SRR934717      RNA-Seq          202  PRJNA212047 SAMN02251223           GEO      public        Diseas
## SRR934718      RNA-Seq          200  PRJNA212047 SAMN02251267           GEO      public        Diseas
## SRR934719      RNA-Seq          202  PRJNA212047 SAMN02251226           GEO      public        Diseas
## SRR934720      RNA-Seq          202  PRJNA212047 SAMN02251225           GEO      public        Diseas
## SRR934721      RNA-Seq          200  PRJNA212047 SAMN02251260           GEO      public        Diseas
## ...                ...          ...          ...          ...           ...         ...            .
## SRR934986      RNA-Seq          202  PRJNA212047 SAMN02251131           GEO      public        Diseas
## SRR934987      RNA-Seq          202  PRJNA212047 SAMN02251128           GEO      public        Diseas
## SRR934988      RNA-Seq          202  PRJNA212047 SAMN02251129           GEO      public        Diseas
## SRR934989      RNA-Seq          202  PRJNA212047 SAMN02251132           GEO      public        Diseas
## SRR934990      RNA-Seq          202  PRJNA212047 SAMN02251137           GEO      public        Diseas
##           insertsize_l        instrument_s librarylayout_s libraryselection_s librarysource_s  loadda
##              <integer>         <character>     <character>        <character>     <character> <chara
## SRR934717            0 Illumina HiSeq 2000          PAIRED               cDNA  TRANSCRIPTOMIC  2013-
## SRR934718            0 Illumina HiSeq 2000          PAIRED               cDNA  TRANSCRIPTOMIC  2013-
## SRR934719            0 Illumina HiSeq 2000          PAIRED               cDNA  TRANSCRIPTOMIC  2013-
## SRR934720            0 Illumina HiSeq 2000          PAIRED               cDNA  TRANSCRIPTOMIC  2013-
## SRR934721            0 Illumina HiSeq 2000          PAIRED               cDNA  TRANSCRIPTOMIC  2013-
## ...                ...                 ...             ...                ...             ...
## SRR934986            0 Illumina HiSeq 2000          PAIRED               cDNA  TRANSCRIPTOMIC  2013-
## SRR934987            0 Illumina HiSeq 2000          PAIRED               cDNA  TRANSCRIPTOMIC  2013-
## SRR934988            0 Illumina HiSeq 2000          PAIRED               cDNA  TRANSCRIPTOMIC  2013-
## SRR934989            0 Illumina HiSeq 2000          PAIRED               cDNA  TRANSCRIPTOMIC  2013-
## SRR934990            0 Illumina HiSeq 2000          PAIRED               cDNA  TRANSCRIPTOMIC  2013-
##            mbytes_l   organism_s  platform_s releasedate_s sample_name_s sra_sample_s sra_study_s sa
##           <integer>  <character> <character>   <character>   <character>  <character> <character>
## SRR934717      3584 Homo sapiens    ILLUMINA    2014-07-21    GSM1185864    SRS457680   SRP027383
## SRR934718      2853 Homo sapiens    ILLUMINA    2014-07-21    GSM1185865    SRS457681   SRP027383
## SRR934719      2650 Homo sapiens    ILLUMINA    2014-07-21    GSM1185866    SRS457682   SRP027383
## SRR934720      3829 Homo sapiens    ILLUMINA    2014-07-21    GSM1185867    SRS457683   SRP027383
## SRR934721      2267 Homo sapiens    ILLUMINA    2014-07-21    GSM1185868    SRS457684   SRP027383
## ...             ...          ...         ...           ...           ...          ...         ...
## SRR934986      2832 Homo sapiens    ILLUMINA    2014-07-21    GSM1186133    SRS457949   SRP027383
## SRR934987      2792 Homo sapiens    ILLUMINA    2014-07-21    GSM1186134    SRS457950   SRP027383
## SRR934988      2822 Homo sapiens    ILLUMINA    2014-07-21    GSM1186135    SRS457951   SRP027383
## SRR934989      3220 Homo sapiens    ILLUMINA    2014-07-21    GSM1186136    SRS457952   SRP027383
## SRR934990      3727 Homo sapiens    ILLUMINA    2014-07-21    GSM1186137    SRS457953   SRP027383
##           development         sex   age_units       age     disease clinical_stage_1
##           <character> <character> <character> <numeric> <character>      <character>
## SRR934717       Adult      female       Years        37       Tumor         Grade II        Oligoden
## SRR934718       Adult        male       Years        25       Tumor         Grade II        Oligoden
```

```
## SRR934719       Adult        male       Years        47       Tumor         Grade II             Olig
## SRR934720       Adult        male       Years        34       Tumor         Grade II        Oligoden
## SRR934721       Adult      female       Years        31       Tumor         Grade II        Oligoden
## ...               ...         ...         ...       ...         ...              ...
## SRR934986       Adult        male       Years        38       Tumor         Grade II        Oligoden
## SRR934987       Adult        male       Years        38       Tumor         Grade II        Oligoden
## SRR934988       Adult        male       Years        55       Tumor         Grade IV
## SRR934989       Adult        male       Years        40       Tumor         Grade II
## SRR934990       Adult        male       Years        36       Tumor        Grade III Anaplastic Olig
##                 pathology clinical_stage_2 present_in_recount
##               <character>      <character>          <logical>
## SRR934717 + IDH1 Mutation               NA               TRUE
## SRR934718 - IDH1 Mutation               NA               TRUE
## SRR934719 + IDH1 Mutation               NA               TRUE
## SRR934720 + IDH1 Mutation               NA               TRUE
## SRR934721               NA               NA               TRUE
## ...                    ...              ...                ...
## SRR934986 - IDH1 Mutation               NA               TRUE
## SRR934987 + IDH1 Mutation        Recurrent               TRUE
## SRR934988 + IDH1 Mutation        Recurrent               TRUE
## SRR934989 - IDH1 Mutation        Recurrent               TRUE
## SRR934990 + IDH1 Mutation               NA               TRUE
```

Several of these variables are technical and may be duplicated with data already present, such as the SRA Experiment ids. We can still use them to verify that entries are correctly matched. Other variables might not be of huge relevance for this study such as `disease_status` since all samples in this study are from diseased tissue. However, they might be useful when working with other studies or doing meta-analyses.

```r
## Check experiment ids
identical(rse_gene$experiment, rse_gene$experiment_s)
```

```
## [1] TRUE
```

```r
## No healthy controls in this study
table(rse_gene$disease_status)
```

```
##
## Disease
##     270
```

```r
## All ages reported in the same unit
table(rse_gene$age_units)
```

```
##
## Years
##   270
```

In this study there are several variables of biological interest that we can use for different analyses. We have information about `sex`, `age`, `tumor_type`, `pathology`, `clinical_stage_1` and `clinical_stage_2`. These variables are described in more detail in the original study (Bao, Chen, Yang, Zhang, et al., 2014). Below we explore each variable at a time, to get an idea on how diverse the data is.

```r
## Univariate exploration of the biological variables for SRP027383
table(rse_gene$sex)
```

```
##
## female   male
##    102    166
```

```r
summary(rse_gene$age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   18.00   36.00   42.00   43.12   51.00   81.00       2
```

```r
table(rse_gene$clinical_stage_1)
```

```
##
##  Grade II Grade III  Grade IV
##        98        72        98
```

```r
table(rse_gene$tumor_type)
```

```
##
##            Anaplastic Astrocytomas Anaplastic Oligodendroastrocytomas      Anaplastic Oligodendrogli
##                                 24                                 35
##                        Astrocytoma                       Glioblastoma             Oligodendroastrocy
##                                 41                                 99
##                  Oligodendroglioma
##                                 21
```

```r
table(rse_gene$pathology, useNA = 'ifany')
```

```
##
## - IDH1 Mutation + IDH1 Mutation            <NA>
##             121             137              12
```

```r
table(rse_gene$clinical_stage_2, useNA = 'ifany')
```

```
##
##   Primary Recurrent Secondary      <NA>
##        59        59        20       132
```

We can ask some questions such as is there a difference in the mean age by sex or if the tumor grade (`clinical_stage_1`), the tumor type or the pathology is associated with sex. The answer is no for these questions so we can infer that the study design is well balanced so far.

```r
## Age mean difference by sex? No
with(colData(rse_gene), t.test(age ~ sex))
```

```
##
##  Welch Two Sample t-test
##
## data:  age by sex
## t = 0.52713, df = 201.03, p-value = 0.5987
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.101339  3.634767
## sample estimates:
## mean in group female   mean in group male
##             43.59804             42.83133
```

```
## Tumor grade and sex association? No
with(colData(rse_gene), addmargins(table(sex, clinical_stage_1)))
```

```
##         clinical_stage_1
## sex      Grade II Grade III Grade IV Sum
##   female       41        27       34 102
##   male         57        45       64 166
##   Sum          98        72       98 268
```

```
with(colData(rse_gene), chisq.test(table(sex, clinical_stage_1)))
```

```
##
##  Pearson's Chi-squared test
##
## data:  table(sex, clinical_stage_1)
## X-squared = 1.0736, df = 2, p-value = 0.5846
```

```
## Tumor type and sex association? No
with(colData(rse_gene), addmargins(table(sex, tumor_type)))
```

```
##         tumor_type
## sex      Anaplastic Astrocytomas Anaplastic Oligodendroastrocytomas Anaplastic Oligodendrogliomas As
##   female                       7                                 18                             2
##   male                        17                                 17                            11
##   Sum                         24                                 35                            13
##         tumor_type
## sex      Glioblastoma Oligodendroastrocytoma Oligodendroglioma Sum
##   female           34                     16                 7 102
##   male             64                     20                14 166
##   Sum              98                     36                21 268
```

```
with(colData(rse_gene), chisq.test(table(sex, tumor_type)))
```

```
## Warning in chisq.test(table(sex, tumor_type)): Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  table(sex, tumor_type)
## X-squared = 8.1801, df = 6, p-value = 0.2252
```

```r
## Sex and pathology association? No
with(colData(rse_gene), addmargins(table(sex, pathology)))
```

```
##         pathology
## sex       - IDH1 Mutation + IDH1 Mutation Sum
##   female               39              59  98
##   male                 82              78 160
##   Sum                 121             137 258
```

```r
with(colData(rse_gene), chisq.test(table(sex, pathology)))
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  table(sex, pathology)
## X-squared = 2.7583, df = 1, p-value = 0.09675
```

Gene differential expression analysis

Gene DE setup

Now that we have sample metadata to work with we can proceed to perform a differential expression analysis at the gene level. To get started we need to load some packages.

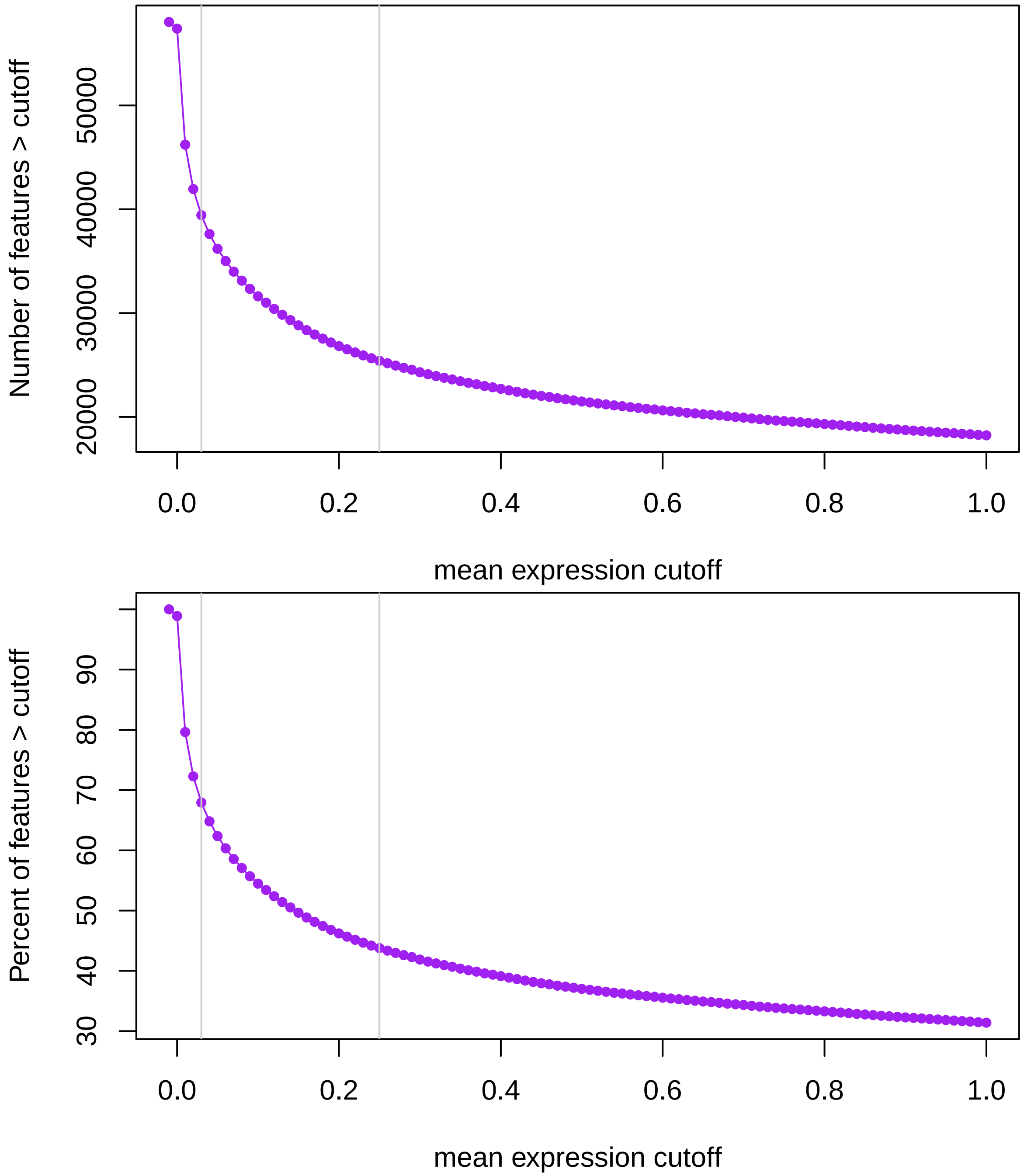
```r
## Load required packages for DE analysis
library('limma')
library('edgeR')
library('jaffelab')
## You can install it with
# devtools::install_github('LieberInstitute/jaffelab')
```

From our earlier exploration, we noticed that not all samples have pathology information, so we will drop those that are missing this information.

```r
## Keep only the samples that have pathology reported
has_patho <- rse_gene[, !is.na(rse_gene$pathology)]
```

Next we will compute RPKM values and use `expression_cutoff()` from *jaffelab* to get a suggested RPKM cutoff for dropping genes with low expression levels. Note that you can also use *genefilter* or other packages for computing a low expression cutoff. Figure @ref(fig:exprcut)A shows the relationship between the mean RPKM cutoff and the number of features above the given cutoff. Figure @ref(fig:exprcut)B is the same information but in percent. Figure @ref(fig:exprcut)C is a tad more complicated as it explore the relationship between the cutoff and the distribution of the number of non-zero samples. All three figures show estimated points where the curves bend and simply provide a guide for choosing a cutoff.

```r
## Compute RPKM and mean RPKM
rpkm <- getRPKM(scale_counts(has_patho))
rpkm_mean <- rowMeans(rpkm)
## Esmate a mean RPKM cutoff
expr_cuts <- expression_cutoff(rpkm)
```



```
## 2020-11-13 16:27:56 the suggested expression cutoff is 0.23
```

```r
round(mean(expr_cuts), 2)
```

```
## [1] 0.23
```

```r
## Filter genes with low levels of expression
has_patho <- has_patho[rpkm_mean > round(mean(expr_cuts), 2), ]
```

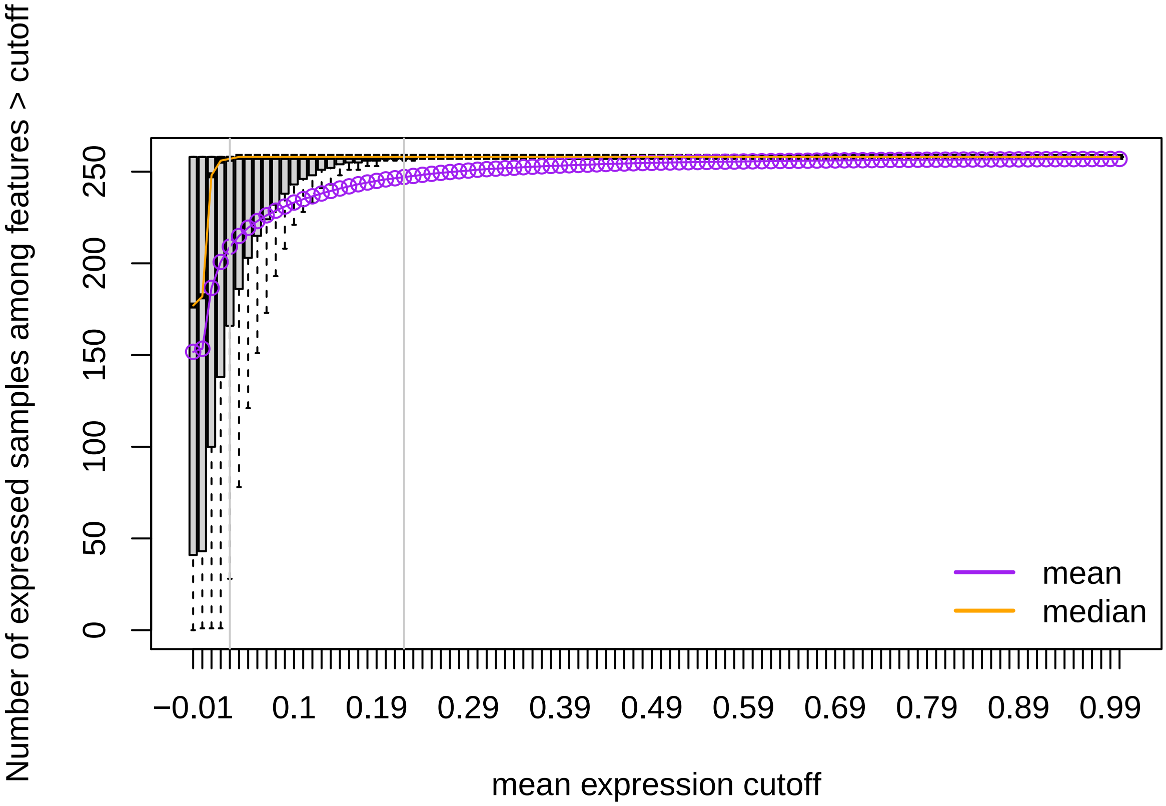


Figure 2: C. Distribution of number of expressed samples across all genes at a given mean RPKM cutoff

Having filtered the genes with low levels of expression, we can now normalize the read counts and identify genes that either have a linear trend or quadratic trend in expression levels between tumor grades II, III and IV while adjusting for age, sex and pathology. Note that this is just an example and you are welcome to try other models. We will use functions from *edgeR* and *limma*.

```
## Get read counts and normalize
dge <- DGEList(counts = assays(scale_counts(has_patho))$counts,
    genes = rowRanges(has_patho))
dge <- calcNormFactors(dge)

## Build the DE model
## See https://support.bioconductor.org/p/54707/ for details
mod <- with(colData(has_patho),
    model.matrix(~ ordered(clinical_stage_1) + sex + age + pathology))

## Terms of the DE model
colnames(mod)
```

```
## [1] "(Intercept)"                  "ordered(clinical_stage_1).L" "ordered(clinical_stage_1).Q"
## [4] "sexmale"                      "age"                         "pathology+ IDH1 Mutation"
```

```
## Check that the dimensions match
stopifnot(ncol(dge) == nrow(mod))

## Run voom then run limma model
```

```
gene_voom <- voom(dge, mod)
gene_fit <- eBayes(lmFit(gene_voom, mod))
```

Now that we have fitted our differential expression model we can find which genes have a linear or a quadratic change in expression along tumor grade progression. At a false discovery rate (FDR) of 1% none of the genes have a quadratic effect.

```
## Extract the stats for both coefficients
stats_linear <- topTable(gene_fit, coef = 2, p.value = 1,
    number = nrow(has_patho), sort.by = 'none')
stats_quad <- topTable(gene_fit, coef = 3, p.value = 1,
    number = nrow(has_patho), sort.by = 'none')

## How many genes are DE for the linear and the quadratic terms at FDR 1%?
addmargins(table('FDR 1% DE linear' = stats_linear$adj.P.Val < 0.01,
    'FDR 1% DE quadractic' = stats_quad$adj.P.Val < 0.01))
```

```
##                  FDR 1% DE quadractic
## FDR 1% DE linear FALSE   Sum
##            FALSE 13343 13343
##            TRUE  12585 12585
##            Sum   25928 25928
```

The fold changes are not necessarily going in the same directions for the differentially expressed genes in the linear term. From the Chi-squared test we can see that the signs are not independent. We could use this information to further explore the gene subsets.

```
## Are the fold changes on the same direction?
addmargins(table(
    'logFC sign linear' = sign(stats_linear$logFC[
        stats_linear$adj.P.Val < 0.01]),
    'logFC sign quadratic' = sign(stats_quad$logFC[
        stats_linear$adj.P.Val < 0.01]))
)
```

```
##                  logFC sign quadratic
## logFC sign linear    -1     1   Sum
##                -1   2766  3816  6582
##                1    3766  2237  6003
##                Sum  6532  6053 12585
```

```
chisq.test(table(
    'logFC sign linear' = sign(stats_linear$logFC[
        stats_linear$adj.P.Val < 0.01]),
    'logFC sign quadratic' = sign(stats_quad$logFC[
        stats_linear$adj.P.Val < 0.01]))
)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  table(‘logFC sign linear‘ = sign(stats_linear$logFC[stats_linear$adj.P.Val <     0.01]), ‘logF
## X-squared = 538.67, df = 1, p-value < 2.2e-16
```

Visualize DE genes

There are thousands of genes that have are differentially expressed in a linear progression of tumor grades. As always, it's always good to visually check some of these genes. For example, we could plot the top 100 DE genes, the 1000 to 1100 top DE genes, etc. The expression can be visualized at different points. We could visualize the raw expression counts (Figure @ref(fig:topgene1)), the voom-normalized expression (Figure @ref(fig:topgene2)) (Law, Chen, Shi, and Smyth, 2014), or the *cleaned* voom-normalized expression (Figure @ref(fig:topgene3)). The last one is the normalized expression where we regress out the effects of the adjustment covariates. This can be done using the `cleaningY()` function from *jaffelab*.

In the following code, we first computed the *cleaned* normalized expression protecting the intercept term as well as the linear and quadratic trend terms. We also write a function that we can use to select which genes to plot as well as actually make the visualization with some nice features (colors, jitter points, linear trend line).

```r
## Regress out sex, age and pathology from the gene expression
cleaned_expr <- cleaningY(gene_voom$E, mod, P = 3)

## gene plotting function
plot_gene <- function(ii, type = 'cleaned', sign = 'any') {
    ## Keep the jitter reproducible
    set.seed(20180203)

    ## Order by FDR and subset by logFC sign if necessary
    if(sign == 'any') {
        fdr_sorted <- with(stats_linear, gene_id[order(adj.P.Val)])
    } else {
        fdr_sorted <- with(stats_linear[sign(stats_linear$logFC) == sign, ],
            gene_id[order(adj.P.Val)])
    }

    ## Get the actual gene it matches originally
    i <- match(fdr_sorted[ii], names(rowRanges(has_patho)))

    ## Define what type of expression we are looking at
    if(type == 'cleaned') {
        y <- cleaned_expr[i, ]
        ylab <- 'Normalized Expr: age, sex, pathology removed'
    } else if (type == 'norm') {
        y <- gene_voom$E[i, ]
        ylab <- 'Normalized Expr'
    } else if (type == 'raw') {
        y <- dge$counts[i, ]
        ylab <- 'Raw Expr'
    }
    ylim <- abs(range(y)) * c(0.95, 1.05) * sign(range(y))

    ## Plot components
    x <- ordered(has_patho$clinical_stage_1)
    title <- with(stats_linear, paste(gene_id[i], symbol[i], 'FDR',
        signif(adj.P.Val[i], 3)))

    ## Make the plot ^^
    plot(y ~ x, xlab = 'Tumor grade', ylab = ylab, outline = FALSE,
        ylim = ylim, main = title)
```

```
    points(y ~ jitter(as.integer(x), 0.6),
        bg = c("#E69F00", "#009E73", "#D55E00")[as.integer(x)], pch = 21)
    abline(lm(y ~ as.integer(x)), lwd = 3, col = "#CC79A7")
}
```

Having built our plotting function, we can now visualize the top gene as shown in Figures @ref(fig:topgene1), @ref(fig:topgene2) and @ref(fig:topgene3). In this case, there's not a large difference between the cleaned expression in Figure @ref(fig:topgene3) and the normalized expression in Figure @ref(fig:topgene2). From GeneCards we can see that the *SMC4* gene plays a role in the structural maintenance of chromosomes, which make sense in our context. Figure @ref(fig:topgene4) shows the top DE gene with a decreasing expression trend across tumor grade progression. *CCNI2* is a paralog of *CCNI* which has been implicated in mitosis.

```
## Visualize the top gene
plot_gene(1, 'raw')
```

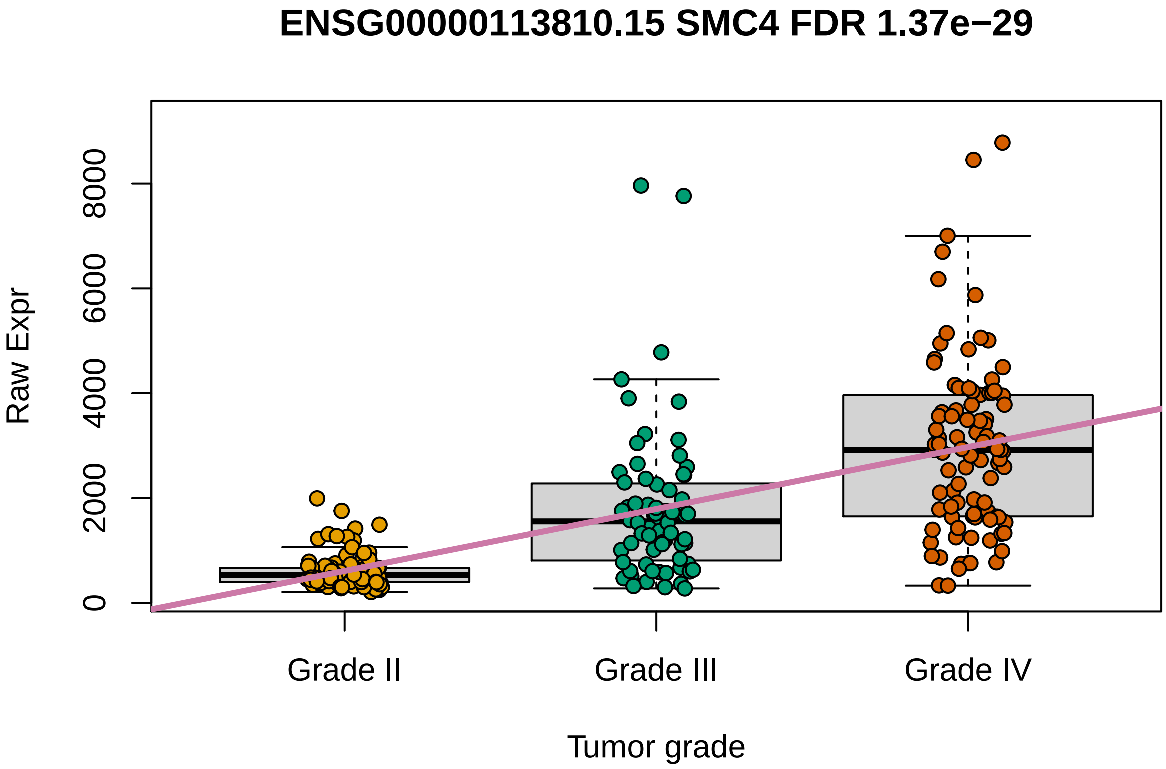


Figure 3: Raw expression for the top DE gene.

```
plot_gene(1, 'norm')
```

```
plot_gene(1)
```

```
## Visualize top gene with a downward trend
plot_gene(1, sign = '-1')
```

We are not experts in gliomas, but maybe your colleagues are and might recognize important genes. You can use the following code to make plots of some of the top DE genes in both directions and share the images with them to get feedback. Check the top50_increasing and top50_decreasing genes in the linked PDF files.

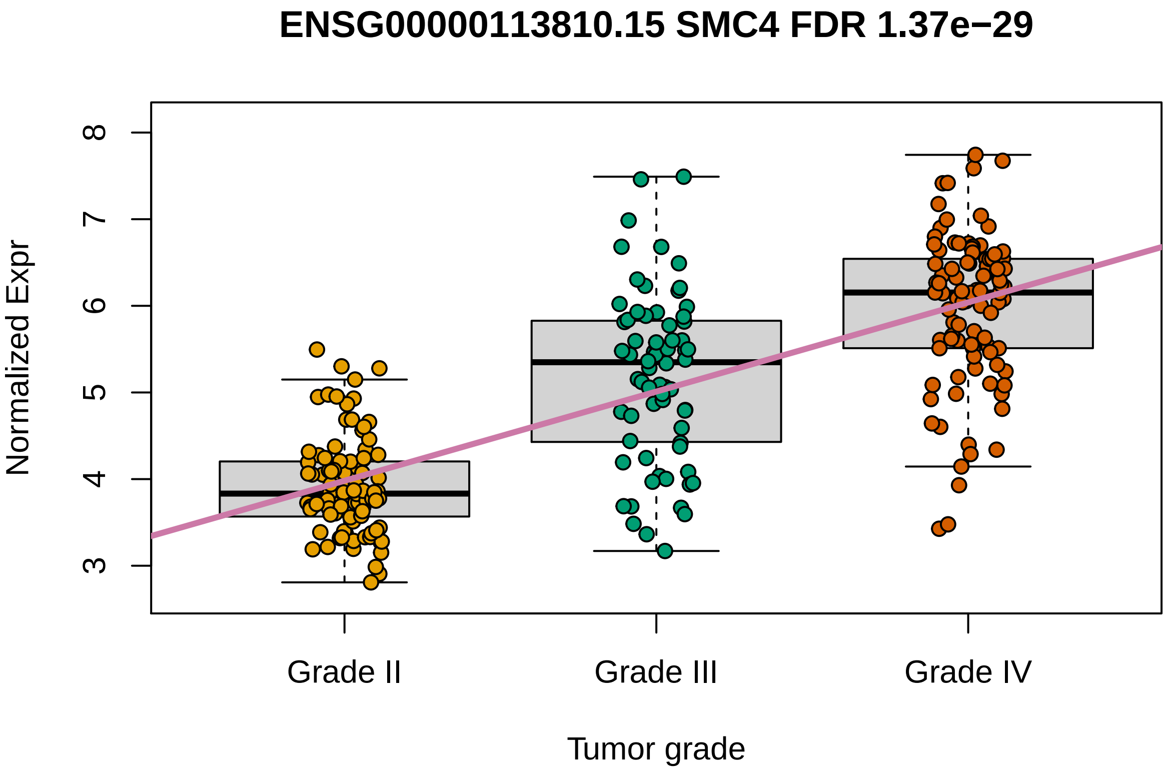


Figure 4: Voom-normalized expression for the top DE gene.

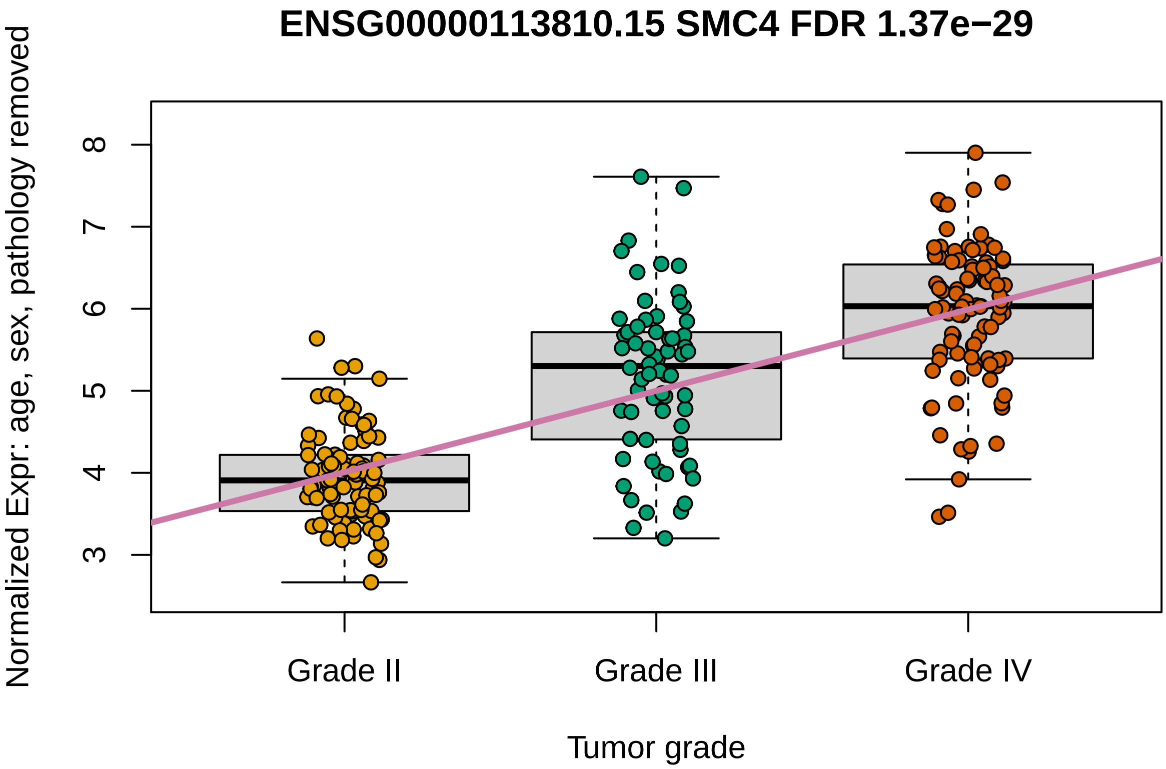


Figure 5: Cleaned voom-normalized expression for the top DE gene.

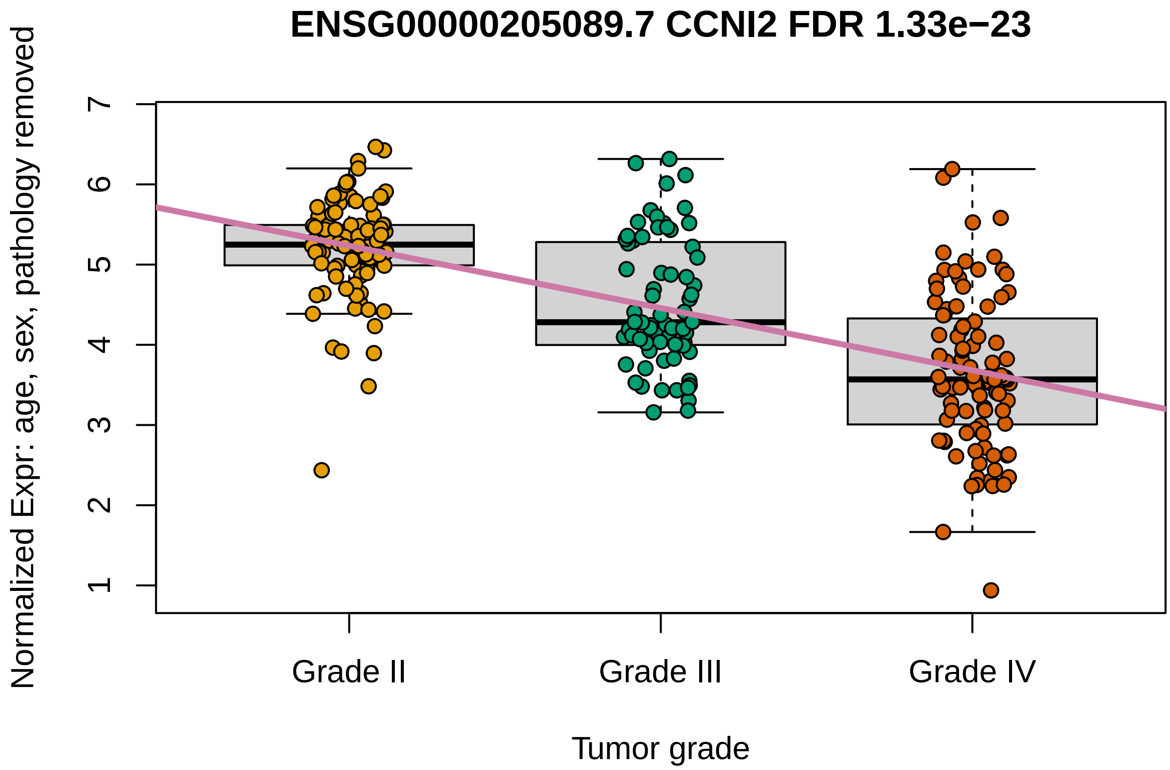


Figure 6: Cleaned voom-normalized expression for the top DE gene with a decreasing trend.

```
## Plot the top 50 increasing and decreasing genes
pdf('top50_increasing.pdf')
for(i in seq_len(50)) plot_gene(i, sign = '1')
dev.off()
```

```
## pdf
##   2
```

```
pdf('top50_decreasing.pdf')
for(i in seq_len(50)) plot_gene(i, sign = '-1')
dev.off()
```

```
## pdf
##   2
```

Gene ontology

Rather than look at the GeneCards for each gene, we can explore which gene ontologies are enriched in the DE genes that have a decreasing and an increasing trend with tumor grade progression. We can use *clusterProfiler* for this exploratory task[2].

---

[2] If you haven't done gene ontology enrichment analyses before check the vignette at bioconductor.org/packages/clusterProfiler.

```r
library('clusterProfiler')
```

We need to extract the gene ids for our sets of genes of interest. Lets explore again the contents of the `stats_linear` object we created earlier. In the `gene_id` column we have the Gencode ids, which can be converted to ENSEMBL gene ids that *clusterProfiler* can then use.

```r
head(stats_linear)
```

```
##                     seqnames     start       end  width strand            gene_id bp_length   symbol
## ENSG00000000003.14     chrX 100627109 100639991  12883      - ENSG00000000003.14      4535   TSPAN6
## ENSG00000000005.5      chrX 100584802 100599885  15084      +  ENSG00000000005.5      1610     TNMD
## ENSG00000000419.12    chr20  50934867  50958555  23689      - ENSG00000000419.12      1207     DPM1
## ENSG00000000457.13     chr1 169849631 169894267  44637      - ENSG00000000457.13      6883    SCYL3
## ENSG00000000460.16     chr1 169662007 169854080 192074      + ENSG00000000460.16      5967 C1orf112
## ENSG00000000938.12     chr1  27612064  27635277  23214      - ENSG00000000938.12      3474      FGR
##                       AveExpr          t      P.Value    adj.P.Val         B
## ENSG00000000003.14  5.402743  2.7864882 5.727935e-03 1.160718e-02 -3.425260
## ENSG00000000005.5  -2.971194 -0.2400606 8.104758e-01 8.521844e-01 -6.221646
## ENSG00000000419.12  4.238778  7.5639336 7.057517e-13 1.211836e-11 18.600807
## ENSG00000000457.13  4.018564  2.7256833 6.860606e-03 1.364229e-02 -3.455390
## ENSG00000000460.16  3.428518  7.1034714 1.206828e-11 1.542170e-10 15.899023
## ENSG00000000938.12  2.899481  2.4728140 1.405680e-02 2.601646e-02 -3.925556
```

With the following code we extract all the DE genes at a FDR of 1% that have an increasing or a decreasing trend. The code comments include a way you could further subset these genes to look at say the top 200 DE genes in each direction. We will use as our *universe* of genes all the genes that passed our low expression filter.

```r
## Get ENSEMBL gene ids for all the DE genes with a decreasing and an
## increasing trend with tumor grade progression
de_genes <- lapply(c('-1', '1'), function(s) {
    ens <- with(stats_linear, gene_id[sign(logFC) == s & adj.P.Val < 0.01])
    ## Code if you wanted the top 200 instead
    #ens <- with(stats_linear[sign(stats_linear$logFC) == s, ],
    #    head(gene_id[order(adj.P.Val)], 200))
    ens <- gsub('\\..*', '', ens)
    return(ens)
})
names(de_genes) <- c('decreasing', 'increasing')
uni <- with(stats_linear, gsub('\\..*', '', gene_id))
```

Now that we have our `list` object with the set of genes with a decreasing or an increasing trend as well as our set of universe genes, we can compare the sets using `compareCluster()`. We will check the biological process, molecular function and cellular component ontologies.

```r
## Which GO terms are enriched?
go_comp <- lapply(c('BP', 'MF', 'CC'), function(bp) {
    message(paste(Sys.time(), 'processing', bp))
    compareCluster(de_genes, fun = "enrichGO",
        universe = uni, OrgDb = 'org.Hs.eg.db',
        ont = bp, pAdjustMethod = "BH",
        pvalueCutoff  = 0.05, qvalueCutoff  = 0.05,
```

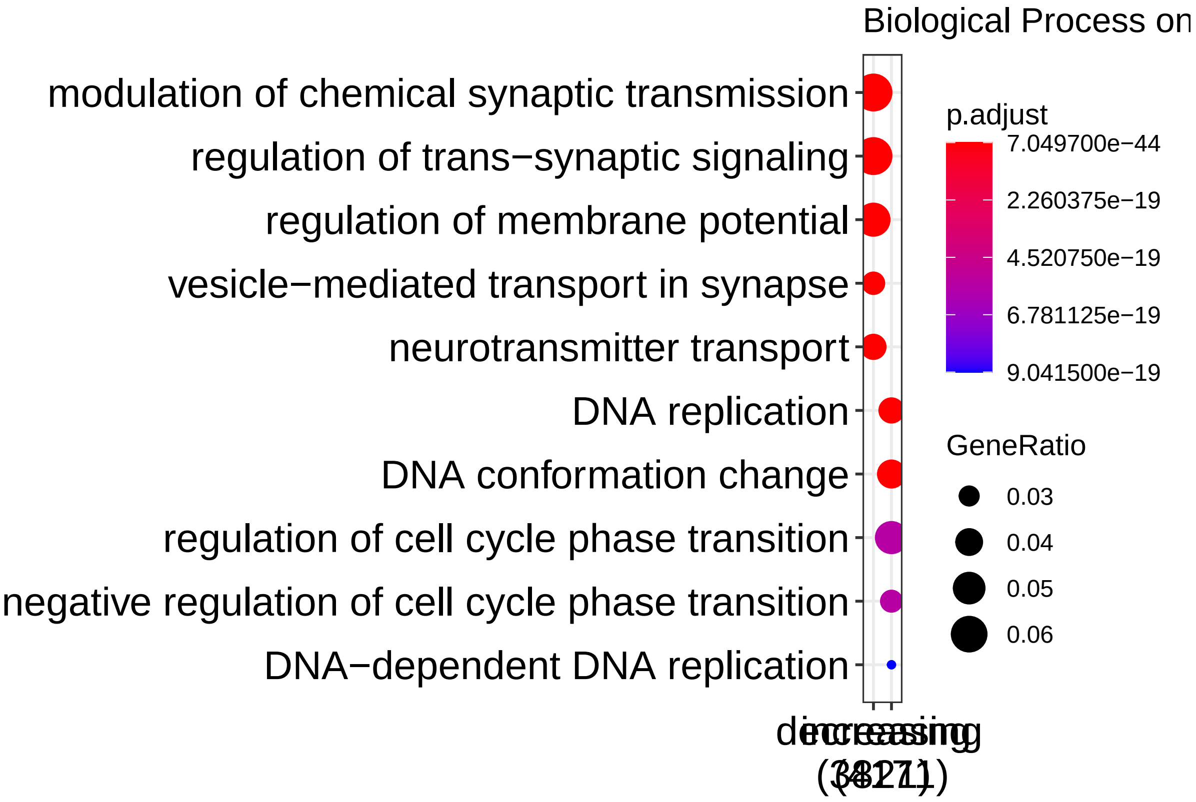
```
        readable= TRUE, keyType = 'ENSEMBL')
})
```

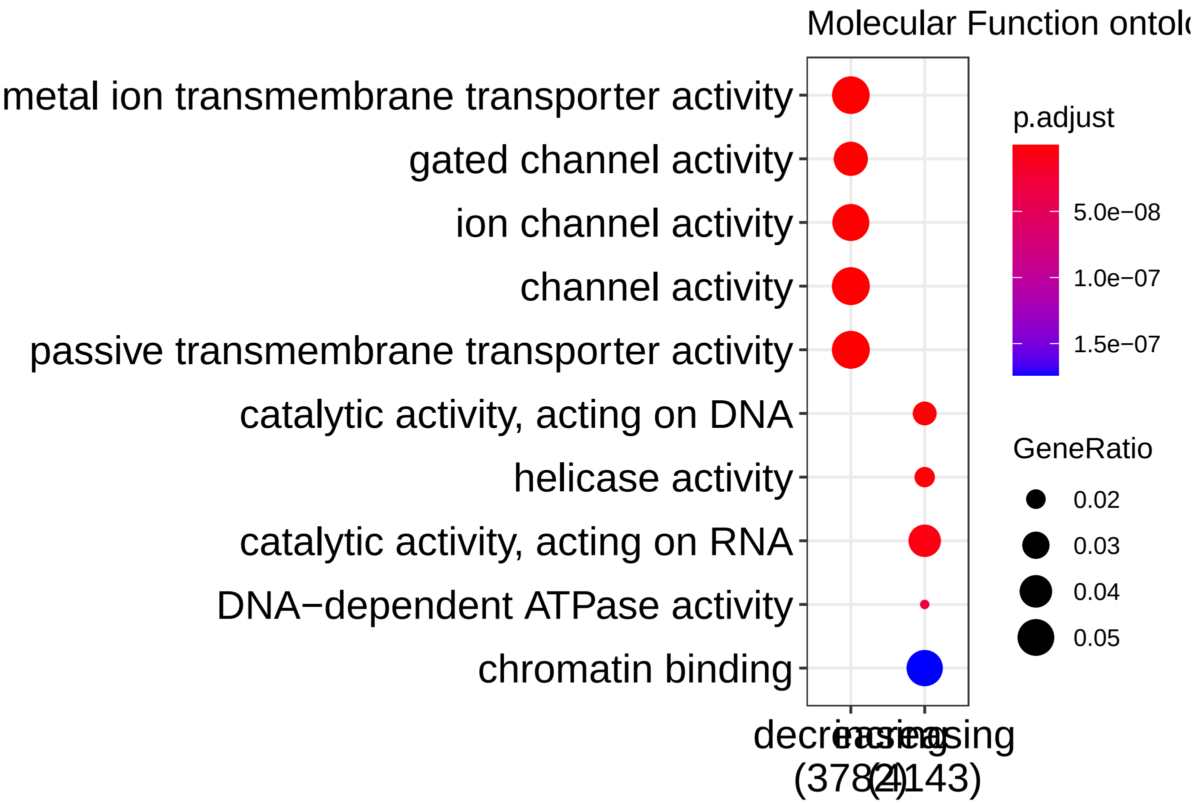
```
## 2020-11-13 13:01:25 processing BP

## 2020-11-13 13:02:43 processing MF

## 2020-11-13 13:03:01 processing CC
```

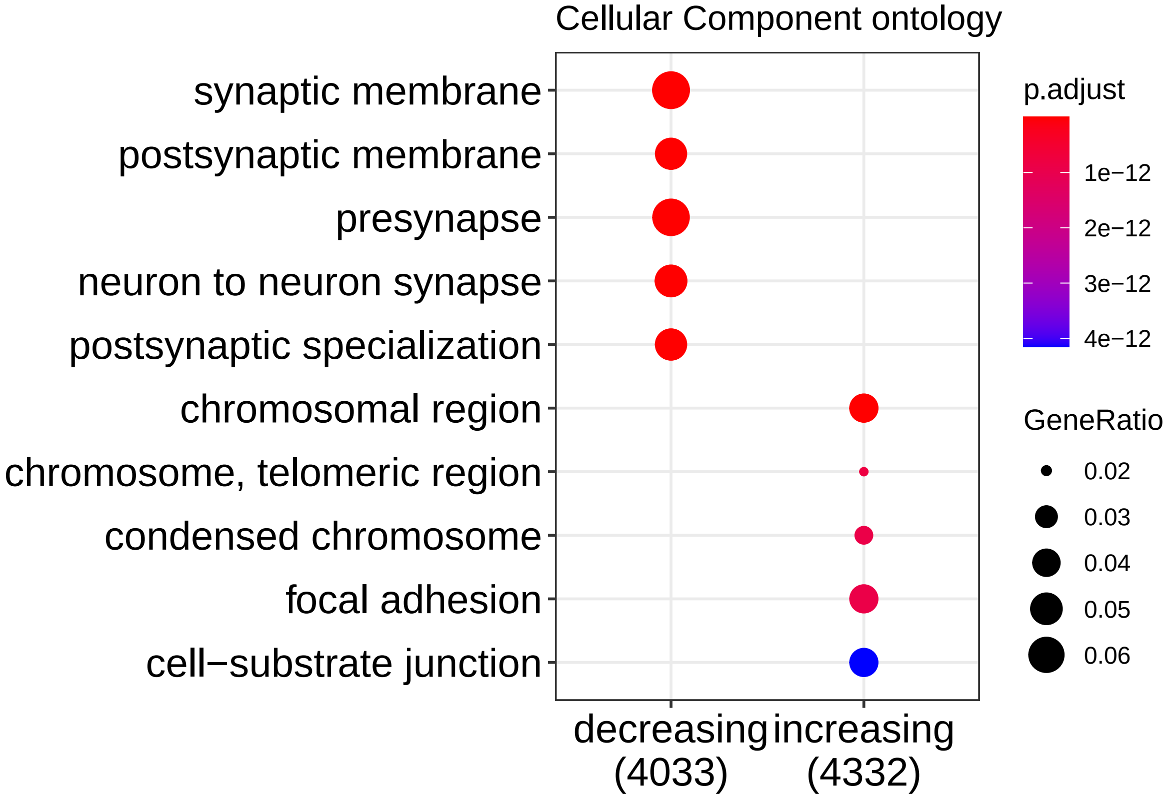
```
names(go_comp) <- c('Biological Process', 'Molecular Function',
    'Cellular Component')
```

Now that we have the data for each of the ontologies we can visualize the results using `clusterProfiler::dotplot()`. Figure @ref(fig:goplot)A shows the enriched biological process terms where we see terms enriched for DNA replication and chromosome segregation in the genes with an increasing expression relationship with grade tumor progression. Intuitively this makes sense since gliomas are a type of cancer. The enriched molecular function ontology terms show in Figure @ref(fig:goplot)B reflect the same picture with transmembrane transporters enriched in the genes with a decreasing expression association with grade tumor progression. Figure @ref(fig:goplot)C shows the enriched cellular components with chromosome-releated terms related with the genes that have a higher expression as tumor progression advances. This is related to the findings in the original study where they focused in gene fusions (Bao, Chen, Yang, Zhang, et al., 2014).

```
## Visualize enriched GO terms
xx <- lapply(names(go_comp), function(bp) {
    print(dotplot(go_comp[[bp]], title = paste(bp, 'ontology'), font.size = 15))
    return(NULL)
})
```







We can finally save our exploratory results in case we want to carry out more analyses with them later on.

```r
## Save results
save(stats_linear, stats_quad, go_comp, file = 'example_results.Rdata')
```

Conclusions

In this document we showed how you can download expression data from `recount2` using the *recount* package and add the sample metadata from `recount_brain`. We then illustrated how both the sample metadata and expression data can be used to explore a biological question of interest. We identified 6582 and 6003 differentially expressed genes at a FDR of 1% with decreasing and increasing linear trends in expression as tumor grade progresses while adjusting for age (in years), sex, and pathology (IDH1 mutation presence/absence).

Reproducibility

```r
## Reproducibility information
Sys.time()
```

```
## [1] "2020-11-13 16:28:32 EST"
```

```r
proc.time()
```

```
##    user  system elapsed 
## 211.309  20.479 295.832
```

```r
options(width = 120)
devtools::session_info()
```

```
## - Session info -------------------------------------------------------------------------------------
##  setting  value                                      
##  version  R version 4.0.2 Patched (2020-06-24 r78746)
##  os       CentOS Linux 7 (Core)                      
##  system   x86_64, linux-gnu                          
##  ui       X11                                        
##  language (EN)                                       
##  collate  en_US.UTF-8                                
##  ctype    en_US.UTF-8                                
##  tz       US/Eastern                                 
##  date     2020-11-13                                 
## 
## - Packages -----------------------------------------------------------------------------------------
##  package              * version  date       lib source                            
##  AnnotationDbi          1.50.3   2020-07-25 [2] Bioconductor                      
##  askpass                1.1      2019-01-13 [2] CRAN (R 4.0.0)                    
##  assertthat             0.2.1    2019-03-21 [2] CRAN (R 4.0.0)                    
##  backports              1.2.0    2020-11-02 [1] CRAN (R 4.0.2)                    
##  base64enc              0.1-3    2015-07-28 [2] CRAN (R 4.0.0)                    
##  bibtex                 0.4.2.3  2020-09-19 [2] CRAN (R 4.0.2)
```

```
##  Biobase              * 2.48.0   2020-04-27 [2] Bioconductor
##  BiocFileCache          1.12.1   2020-08-04 [2] Bioconductor
##  BiocGenerics         * 0.34.0   2020-04-27 [2] Bioconductor
##  BiocManager            1.30.10  2019-11-16 [2] CRAN (R 4.0.0)
##  BiocParallel           1.22.0   2020-04-27 [2] Bioconductor
##  BiocStyle            * 2.16.1   2020-09-25 [1] Bioconductor
##  biomaRt                2.44.4   2020-10-13 [2] Bioconductor
##  Biostrings             2.56.0   2020-04-27 [2] Bioconductor
##  bit                    4.0.4    2020-08-04 [2] CRAN (R 4.0.2)
##  bit64                  4.0.5    2020-08-30 [2] CRAN (R 4.0.2)
##  bitops                 1.0-6    2013-08-17 [2] CRAN (R 4.0.0)
##  blob                   1.2.1    2020-01-20 [2] CRAN (R 4.0.0)
##  bookdown               0.21     2020-10-13 [1] CRAN (R 4.0.2)
##  broom                  0.7.2    2020-10-20 [2] CRAN (R 4.0.2)
##  BSgenome               1.56.0   2020-04-27 [2] Bioconductor
##  bumphunter             1.30.0   2020-04-27 [2] Bioconductor
##  callr                  3.5.1    2020-10-13 [2] CRAN (R 4.0.2)
##  cellranger             1.1.0    2016-07-27 [2] CRAN (R 4.0.0)
##  checkmate              2.0.0    2020-02-06 [2] CRAN (R 4.0.0)
##  cli                    2.1.0    2020-10-12 [2] CRAN (R 4.0.2)
##  cluster                2.1.0    2019-06-19 [3] CRAN (R 4.0.2)
##  clusterProfiler      * 3.16.1   2020-08-18 [1] Bioconductor
##  codetools              0.2-16   2018-12-24 [3] CRAN (R 4.0.2)
##  colorspace             1.4-1    2019-03-18 [2] CRAN (R 4.0.0)
##  cowplot                1.1.0    2020-09-08 [1] CRAN (R 4.0.2)
##  crayon                 1.3.4    2017-09-16 [2] CRAN (R 4.0.0)
##  curl                   4.3      2019-12-02 [2] CRAN (R 4.0.0)
##  data.table             1.13.2   2020-10-19 [2] CRAN (R 4.0.2)
##  DBI                    1.1.0    2019-12-15 [2] CRAN (R 4.0.0)
##  dbplyr                 2.0.0    2020-11-03 [1] CRAN (R 4.0.2)
##  DelayedArray         * 0.14.1   2020-07-14 [2] Bioconductor
##  derfinder              1.22.0   2020-04-27 [2] Bioconductor
##  derfinderHelper        1.22.0   2020-04-27 [2] Bioconductor
##  desc                   1.2.0    2018-05-01 [2] CRAN (R 4.0.0)
##  devtools             * 2.3.2    2020-09-18 [2] CRAN (R 4.0.2)
##  digest                 0.6.27   2020-10-24 [1] CRAN (R 4.0.2)
##  DO.db                  2.9      2020-08-06 [1] Bioconductor
##  doRNG                  1.8.2    2020-01-27 [2] CRAN (R 4.0.0)
##  DOSE                   3.14.0   2020-04-27 [1] Bioconductor
##  downloader             0.4      2015-07-09 [2] CRAN (R 4.0.0)
##  dplyr                * 1.0.2    2020-08-18 [2] CRAN (R 4.0.2)
##  edgeR                * 3.30.3   2020-06-02 [2] Bioconductor
##  ellipsis               0.3.1    2020-05-15 [2] CRAN (R 4.0.0)
##  enrichplot             1.8.1    2020-04-29 [1] Bioconductor
##  europepmc              0.4      2020-05-31 [1] CRAN (R 4.0.2)
##  evaluate               0.14     2019-05-28 [2] CRAN (R 4.0.0)
##  fansi                  0.4.1    2020-01-08 [2] CRAN (R 4.0.0)
##  farver                 2.0.3    2020-01-16 [2] CRAN (R 4.0.0)
##  fastmatch              1.1-0    2017-01-28 [1] CRAN (R 4.0.2)
##  fgsea                  1.14.0   2020-04-27 [1] Bioconductor
##  forcats              * 0.5.0    2020-03-01 [2] CRAN (R 4.0.0)
##  foreach                1.5.1    2020-10-15 [2] CRAN (R 4.0.2)
##  foreign                0.8-80   2020-05-24 [3] CRAN (R 4.0.2)
##  Formula                1.2-4    2020-10-16 [2] CRAN (R 4.0.2)
```

```
##  fs                     1.5.0    2020-07-31 [1] CRAN (R 4.0.2)
##  generics               0.1.0    2020-10-31 [1] CRAN (R 4.0.2)
##  GenomeInfoDb         * 1.24.2   2020-06-15 [2] Bioconductor
##  GenomeInfoDbData       1.2.3    2020-05-18 [2] Bioconductor
##  GenomicAlignments      1.24.0   2020-04-27 [2] Bioconductor
##  GenomicFeatures        1.40.1   2020-07-08 [2] Bioconductor
##  GenomicFiles           1.24.0   2020-04-27 [2] Bioconductor
##  GenomicRanges        * 1.40.0   2020-04-27 [2] Bioconductor
##  GEOquery               2.56.0   2020-04-27 [2] Bioconductor
##  ggforce                0.3.2    2020-06-23 [2] CRAN (R 4.0.2)
##  ggplot2              * 3.3.2    2020-06-19 [2] CRAN (R 4.0.2)
##  ggplotify              0.0.5    2020-03-12 [1] CRAN (R 4.0.2)
##  ggraph                 2.0.3    2020-05-20 [2] CRAN (R 4.0.2)
##  ggrepel                0.8.2    2020-03-08 [2] CRAN (R 4.0.0)
##  ggridges               0.5.2    2020-01-12 [1] CRAN (R 4.0.2)
##  glue                   1.4.2    2020-08-27 [1] CRAN (R 4.0.2)
##  GO.db                  3.11.4   2020-10-23 [2] Bioconductor
##  googledrive            1.0.1    2020-05-05 [1] CRAN (R 4.0.0)
##  GOSemSim               2.14.2   2020-09-04 [1] Bioconductor
##  graphlayouts           0.7.1    2020-10-26 [1] CRAN (R 4.0.2)
##  gridExtra              2.3      2017-09-09 [2] CRAN (R 4.0.0)
##  gridGraphics           0.5-0    2020-02-25 [1] CRAN (R 4.0.2)
##  gtable                 0.3.0    2019-03-25 [2] CRAN (R 4.0.0)
##  haven                  2.3.1    2020-06-01 [2] CRAN (R 4.0.2)
##  highr                  0.8      2019-03-20 [2] CRAN (R 4.0.0)
##  Hmisc                  4.4-1    2020-08-10 [2] CRAN (R 4.0.2)
##  hms                    0.5.3    2020-01-08 [2] CRAN (R 4.0.0)
##  htmlTable              2.1.0    2020-09-16 [2] CRAN (R 4.0.2)
##  htmltools              0.5.0    2020-06-16 [2] CRAN (R 4.0.2)
##  htmlwidgets            1.5.2    2020-10-03 [2] CRAN (R 4.0.2)
##  httr                   1.4.2    2020-07-20 [2] CRAN (R 4.0.2)
##  igraph                 1.2.6    2020-10-06 [2] CRAN (R 4.0.2)
##  IRanges              * 2.22.2   2020-05-21 [2] Bioconductor
##  iterators              1.0.13   2020-10-15 [2] CRAN (R 4.0.2)
##  jaffelab             * 0.99.30  2020-06-25 [1] Github (LieberInstitute/jaffelab@42637ff)
##  jpeg                   0.1-8.1  2019-10-24 [2] CRAN (R 4.0.0)
##  jsonlite               1.7.1    2020-09-07 [2] CRAN (R 4.0.2)
##  knitcitations        * 1.0.10   2019-09-15 [1] CRAN (R 4.0.2)
##  knitr                  1.30     2020-09-22 [1] CRAN (R 4.0.2)
##  labeling               0.4.2    2020-10-20 [2] CRAN (R 4.0.2)
##  lattice                0.20-41  2020-04-02 [3] CRAN (R 4.0.2)
##  latticeExtra           0.6-29   2019-12-19 [2] CRAN (R 4.0.0)
##  lifecycle              0.2.0    2020-03-06 [2] CRAN (R 4.0.0)
##  limma                * 3.44.3   2020-06-12 [2] Bioconductor
##  locfit                 1.5-9.4  2020-03-25 [2] CRAN (R 4.0.0)
##  lubridate              1.7.9    2020-06-08 [1] CRAN (R 4.0.0)
##  magick                 2.5.2    2020-11-10 [1] CRAN (R 4.0.2)
##  magrittr               1.5      2014-11-22 [2] CRAN (R 4.0.0)
##  MASS                   7.3-51.6 2020-04-26 [3] CRAN (R 4.0.2)
##  Matrix                 1.2-18   2019-11-27 [3] CRAN (R 4.0.2)
##  matrixStats          * 0.57.0   2020-09-25 [2] CRAN (R 4.0.2)
##  memoise                1.1.0    2017-04-21 [2] CRAN (R 4.0.0)
##  modelr                 0.1.8    2020-05-19 [1] CRAN (R 4.0.0)
##  munsell                0.5.0    2018-06-12 [2] CRAN (R 4.0.0)
```

```
##  nnet                    7.3-14   2020-04-26 [3] CRAN (R 4.0.2)
##  openssl                 1.4.3    2020-09-18 [2] CRAN (R 4.0.2)
##  pillar                  1.4.6    2020-07-10 [2] CRAN (R 4.0.2)
##  pkgbuild                1.1.0    2020-07-13 [2] CRAN (R 4.0.2)
##  pkgconfig               2.0.3    2019-09-22 [2] CRAN (R 4.0.0)
##  pkgload                 1.1.0    2020-05-29 [2] CRAN (R 4.0.2)
##  plyr                    1.8.6    2020-03-03 [2] CRAN (R 4.0.0)
##  png                     0.1-7    2013-12-03 [2] CRAN (R 4.0.0)
##  polyclip                1.10-0   2019-03-14 [2] CRAN (R 4.0.0)
##  prettyunits             1.1.1    2020-01-24 [2] CRAN (R 4.0.0)
##  processx                3.4.4    2020-09-03 [2] CRAN (R 4.0.2)
##  progress                1.2.2    2019-05-16 [2] CRAN (R 4.0.0)
##  ps                      1.4.0    2020-10-07 [2] CRAN (R 4.0.2)
##  purrr                 * 0.3.4    2020-04-17 [2] CRAN (R 4.0.0)
##  qvalue                  2.20.0   2020-04-27 [2] Bioconductor
##  R6                      2.5.0    2020-10-28 [1] CRAN (R 4.0.2)
##  rafalib               * 1.0.0    2015-08-09 [1] CRAN (R 4.0.0)
##  rappdirs                0.3.1    2016-03-28 [2] CRAN (R 4.0.0)
##  RColorBrewer            1.1-2    2014-12-07 [2] CRAN (R 4.0.0)
##  Rcpp                    1.0.5    2020-07-06 [2] CRAN (R 4.0.2)
##  RCurl                   1.98-1.2 2020-04-18 [2] CRAN (R 4.0.0)
##  readr                 * 1.4.0    2020-10-05 [2] CRAN (R 4.0.2)
##  readxl                  1.3.1    2019-03-13 [2] CRAN (R 4.0.0)
##  recount               * 1.14.0   2020-04-27 [2] Bioconductor
##  RefManageR              1.2.12   2019-04-03 [1] CRAN (R 4.0.2)
##  remotes                 2.2.0    2020-07-21 [2] CRAN (R 4.0.2)
##  rentrez                 1.2.2    2019-05-02 [2] CRAN (R 4.0.0)
##  reprex                  0.3.0    2019-05-16 [1] CRAN (R 4.0.0)
##  reshape2                1.4.4    2020-04-09 [2] CRAN (R 4.0.0)
##  rlang                   0.4.8    2020-10-08 [1] CRAN (R 4.0.2)
##  rmarkdown             * 2.5      2020-10-21 [1] CRAN (R 4.0.2)
##  rngtools                1.5      2020-01-23 [2] CRAN (R 4.0.0)
##  rpart                   4.1-15   2019-04-12 [3] CRAN (R 4.0.2)
##  rprojroot               1.3-2    2018-01-03 [2] CRAN (R 4.0.0)
##  Rsamtools               2.4.0    2020-04-27 [2] Bioconductor
##  RSQLite                 2.2.1    2020-09-30 [2] CRAN (R 4.0.2)
##  rstudioapi              0.11     2020-02-07 [2] CRAN (R 4.0.0)
##  rtracklayer             1.48.0   2020-04-27 [2] Bioconductor
##  rvcheck                 0.1.8    2020-03-01 [1] CRAN (R 4.0.2)
##  rvest                   0.3.6    2020-07-25 [2] CRAN (R 4.0.2)
##  S4Vectors             * 0.26.1   2020-05-16 [2] Bioconductor
##  scales                  1.1.1    2020-05-11 [2] CRAN (R 4.0.0)
##  scatterpie              0.1.5    2020-09-09 [1] CRAN (R 4.0.2)
##  segmented               1.3-0    2020-10-27 [1] CRAN (R 4.0.2)
##  sessioninfo           * 1.1.1    2018-11-05 [2] CRAN (R 4.0.0)
##  stringi                 1.5.3    2020-09-09 [2] CRAN (R 4.0.2)
##  stringr               * 1.4.0    2019-02-10 [2] CRAN (R 4.0.0)
##  SummarizedExperiment  * 1.18.2   2020-07-09 [2] Bioconductor
##  survival                3.2-3    2020-06-13 [3] CRAN (R 4.0.2)
##  testthat                3.0.0    2020-10-31 [1] CRAN (R 4.0.2)
##  tibble                * 3.0.4    2020-10-12 [2] CRAN (R 4.0.2)
##  tidygraph               1.2.0    2020-05-12 [2] CRAN (R 4.0.0)
##  tidyr                 * 1.1.2    2020-08-27 [2] CRAN (R 4.0.2)
##  tidyselect              1.1.0    2020-05-11 [2] CRAN (R 4.0.0)
```

```
##  tidyverse           * 1.3.0    2019-11-21 [1] CRAN (R 4.0.0)
##  triebeard             0.3.0    2016-08-04 [1] CRAN (R 4.0.2)
##  tweenr                1.0.1    2018-12-14 [2] CRAN (R 4.0.0)
##  urltools              1.7.3    2019-04-14 [1] CRAN (R 4.0.2)
##  usethis             * 1.6.3    2020-09-17 [2] CRAN (R 4.0.2)
##  VariantAnnotation     1.34.0   2020-04-27 [2] Bioconductor
##  vctrs                 0.3.4    2020-08-29 [1] CRAN (R 4.0.2)
##  viridis               0.5.1    2018-03-29 [2] CRAN (R 4.0.0)
##  viridisLite           0.3.0    2018-02-01 [2] CRAN (R 4.0.0)
##  withr                 2.3.0    2020-09-22 [2] CRAN (R 4.0.2)
##  xfun                  0.19     2020-10-30 [1] CRAN (R 4.0.2)
##  XML                   3.99-0.5 2020-07-23 [2] CRAN (R 4.0.2)
##  xml2                  1.3.2    2020-04-23 [2] CRAN (R 4.0.0)
##  XVector               0.28.0   2020-04-27 [2] Bioconductor
##  yaml                  2.2.1    2020-02-01 [2] CRAN (R 4.0.0)
##  zlibbioc              1.34.0   2020-04-27 [2] Bioconductor
##
## [1] /users/neagles/R/4.0
## [2] /jhpce/shared/jhpce/core/conda/miniconda3-4.6.14/envs/svnR-4.0/R/4.0/lib64/R/site-library
## [3] /jhpce/shared/jhpce/core/conda/miniconda3-4.6.14/envs/svnR-4.0/R/4.0/lib64/R/library
```

References

The analyses were made possible thanks to:

- R (R Core Team, 2020)
- *BiocStyle* (Oleś, Morgan, and Huber, 2020)
- *clusterProfiler* (Yu, Wang, Han, and He, 2012)
- *devtools* (Wickham, Hester, and Chang, 2020)
- *edgeR* (Robinson, McCarthy, and Smyth, 2010; McCarthy, Chen, and Smyth, 2012)
- *jaffelab* (Collado-Torres, Jaffe, and Burke, 2019)
- *knitcitations* (Boettiger, 2019)
- *knitr* (Xie, 2014)
- *limma* (Ritchie, Phipson, Wu, Hu, et al., 2015; Law, Chen, Shi, and Smyth, 2014)
- *recount* (Collado-Torres, Nellore, Kammers, Ellis, et al., 2017; Collado-Torres, Nellore, and Jaffe, 2017)
- *rmarkdown* (Allaire, Xie, McPherson, Luraschi, et al., 2020)

Full bibliography file.

[1] J. Allaire, Y. Xie, J. McPherson, J. Luraschi, et al. rmarkdown: Dynamic Documents for R. R package version 2.5. 2020. URL: https://github.com/rstudio/rmarkdown.

[2] Z. Bao, H. Chen, M. Yang, C. Zhang, et al. "RNA-seq of 272 gliomas revealed a novel, recurrentPTPRZ1-METfusion transcript in secondary glioblastomas". In: Genome Research 24.11 (Aug. 2014), pp. 1765–1773. DOI: 10.1101/gr.165126.113. URL: https://doi.org/10.1101/gr.165126.113.

[3] C. Boettiger. knitcitations: Citations for 'Knitr' Markdown Files. R package version 1.0.10. 2019. URL: https://CRAN.R-project.org/package=knitcitations.

[4] L. Collado-Torres, A. E. Jaffe, and E. E. Burke. jaffelab: Commonly used functions by the Jaffe lab. R package version 0.99.30. 2019. URL: https://github.com/LieberInstitute/jaffelab.

[5] L. Collado-Torres, A. Nellore, and A. E. Jaffe. "recount workflow: Accessing over 70,000 human RNA-seq samples with Bioconductor [version 1; referees: 1 approved, 2 approved with reservations]". In:

F1000Research (2017). DOI: 10.12688/f1000research.12223.1. URL: https://f1000research.com/articles/6-1558/v1.

[6] L. Collado-Torres, A. Nellore, K. Kammers, S. E. Ellis, et al. “Reproducible RNA-seq analysis using recount2”. In: Nature Biotechnology (2017). DOI: 10.1038/nbt.3838. URL: http://www.nature.com/nbt/journal/v35/n4/full/n

[7] C. Law, Y. Chen, W. Shi, and G. Smyth. “Voom: precision weights unlock linear model analysis tools for RNA-seq read counts”. In: Genome Biology 15 (2014), p. R29.

[8] D. J. McCarthy, Y. Chen, and G. K. Smyth. “Differential expression analysis of multifactor RNA-Seq experiments with respect to biological variation”. In: Nucleic Acids Research 40.10 (2012), pp. 4288-4297. DOI: 10.1093/nar/gks042.

[9] A. Oleś, M. Morgan, and W. Huber. BiocStyle: Standard styles for vignettes and other Bioconductor documents. R package version 2.16.1. 2020. URL: https://github.com/Bioconductor/BiocStyle.

[10] R Core Team. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing. Vienna, Austria, 2020. URL: https://www.R-project.org/.

[11] M. E. Ritchie, B. Phipson, D. Wu, Y. Hu, et al. “limma powers differential expression analyses for RNA-sequencing and microarray studies”. In: Nucleic Acids Research 43.7 (2015), p. e47. DOI: 10.1093/nar/gkv007.

[12] M. D. Robinson, D. J. McCarthy, and G. K. Smyth. “edgeR: a Bioconductor package for differential expression analysis of digital gene expression data”. In: Bioinformatics 26.1 (2010), pp. 139-140. DOI: 10.1093/bioinformatics/btp616.

[13] H. Wickham, J. Hester, and W. Chang. devtools: Tools to Make Developing R Packages Easier. R package version 2.3.2. 2020. URL: https://CRAN.R-project.org/package=devtools.

[14] Y. Xie. “knitr: A Comprehensive Tool for Reproducible Research in R”. In: Implementing Reproducible Computational Research. Ed. by V. Stodden, F. Leisch and R. D. Peng. ISBN 978-1466561595. Chapman and Hall/CRC, 2014. URL: http://www.crcpress.com/product/isbn/9781466561595.

[15] G. Yu, L. Wang, Y. Han, and Q. He. “clusterProfiler: an R package for comparing biological themes among gene clusters”. In: OMICS: A Journal of Integrative Biology 16.5 (2012), pp. 284-287. DOI: 10.1089/omi.2011.0118.