

# Package ‘blacksheepr’

October 18, 2024

**Type** Package

**Title** Outlier Analysis for pairwise differential comparison

**Version** 1.19.0

**Description** Blacksheep is a tool designed for outlier analysis in the context of pairwise comparisons in an effort to find distinguishing characteristics from two groups. This tool was designed to be applied for biological applications such as phosphoproteomics or transcriptomics, but it can be used for any data that can be represented by a 2D table, and has two sub populations within the table to compare.

**License** MIT + file LICENSE

**Encoding** UTF-8

**VignetteBuilder** knitr

**RoxygenNote** 6.1.1

**Imports** grid, stats, grDevices, utils, circlize, viridis,  
RColorBrewer, ComplexHeatmap, SummarizedExperiment, pasilla

**Suggests** testthat (>= 2.1.0), knitr, BiocStyle, rmarkdown, curl

**Depends** R (>= 3.6)

**biocViews** Sequencing, RNASeq, GeneExpression, Transcription,  
DifferentialExpression, Transcriptomics

**BugReports** <https://github.com/ruggleslab/blacksheepr/issues>

**git\_url** <https://git.bioconductor.org/packages/blacksheepr>

**git\_branch** devel

**git\_last\_commit** 2a4087a

**git\_last\_commit\_date** 2024-04-30

**Repository** Bioconductor 3.20

**Date/Publication** 2024-10-18

**Author** MacIntosh Cornwell [aut],  
RugglesLab [cre]

**Maintainer** RugglesLab <ruggleslab@gmail.com>

## Contents

annotationlist_builder . . . . .	2
comparison_groupings . . . . .	3
count_outliers . . . . .	3
create_heatmap . . . . .	4
deva . . . . .	5
deva_normalization . . . . .	6
deva_results . . . . .	7
make_comparison_columns . . . . .	8
make_outlier_table . . . . .	8
outlier_analysis . . . . .	9
outlier_heatmap . . . . .	10
sample_annotationdata . . . . .	11
sample_phosphodata . . . . .	12
sample_rndata . . . . .	14
<b>Index</b>	<b>17</b>

---

annotationlist\_builder

*Create the annotation object for plotting in a heatmap*

---

### Description

Create the annotation object for plotting in a heatmap

### Usage

```
annotationlist_builder(metatable, customcolorlist = NULL)
```

### Arguments

metatable        the metatable containing information for the columns  
 customcolorlist  
                   DEFAULT: NULL, enter colorlist to manually set colors

### Value

return the annotation object

### Examples

```
metatable <- data.frame(row.names = c("samp1", "samp2", "samp3", "samp4"),
  A = c(rep("high", 2), rep("low", 2)), B = seq(1,7,2))
customcolorlist <- list(A = c("high" = "red", "low" = "blue"),
  B = circlize::colorRamp2(seq(-5, 5, length = 3),
  RColorBrewer::brewer.pal(3, "Reds")))
annotationlist_builder(metatable, customcolorlist)
```

---

comparison\_groupings    *Create all of the groups based on the input metadata*

---

### Description

Create all of the groups based on the input metadata

### Usage

```
comparison_groupings(comptable)
```

### Arguments

comptable            table where each column will have comparisons drawn from it

### Value

a list with each of the groups as an entry in the list NOTE - this list will be ncol\*2 long where ncol is the number comparisons

### Examples

```
data("sample_annotationdata")
groupings <- comparison_groupings(sample_annotationdata)
```

---

count\_outliers            *Count up the outlier information for each of the groups you have made. If aggregating then you will have to turn the parameter on, but you still input the outlier table. Aggregate will count the total number of outliers AND nonoutliers in its operation, so it needs the original outlier table made by the <make\_outlier\_table> function.*

---

### Description

Count up the outlier information for each of the groups you have made. If aggregating then you will have to turn the parameter on, but you still input the outlier table. Aggregate will count the total number of outliers AND nonoutliers in its operation, so it needs the original outlier table made by the <make\_outlier\_table> function.

### Usage

```
count_outliers(groupings, outlier tab,
  aggregate_features = FALSE, feature_delineator = "\\.")
```

**Arguments**

groupings	table generated by the comparison_groupings function
outliertab	outlier table generated by make_outlier_table
aggregate_features	DEFAULT: FALSE; Toggle the Aggregate feature, which will aggregate features in your table based on the given delineator. Aggregation will output counts for the TOTAL number of outliers and non- outliers across ALL sites you aggregate across.
feature_delineator	DEFAULT: <"\">; What character delineates the separation between primary and secondary features. NOTE: to use proper R syntax with escape characters if necessary Ex) Protein1.Phosphosite1 uses "\" to aggregate on Protein1

**Value**

the tabulated information of outliers per group

**Examples**

```
data("sample_phosphodata")
reftable_function_out <- make_outlier_table(sample_phosphodata[1:1000,])
outliertab <- reftable_function_out$outliertab

data("sample_annotationdata")
groupings <- comparison_groupings(sample_annotationdata)

count_outliers_out <- count_outliers(groupings, outliertab,
  aggregate_features = FALSE)
grouptablist <- count_outliers_out$grouptablist
fractiontab <- count_outliers_out$fractiontab
```

---

create\_heatmap

*Plot out a heatmap*

---

**Description**

Plot out a heatmap

**Usage**

```
create_heatmap(counttab = counttab,
  colmetatable = NULL, colannotationlist = NULL,
  colclusterparam = FALSE, rowclusterparam = FALSE,
  nameparam)
```

**Arguments**

counttab	table with counts, samples -x-axis, features -y-axis
colmetatable	the metatable containing information for the columns
colannotationlist	annotation table for columns, based off colmetatable

```

colclusterparam      cluster the columns?
rowclusterparam      cluster the rows?
nameparam            the title on the heatmap

```

### Value

prints a pdf heatmap out to the designated outpath

### Examples

```

data("sample_phosphodata")
counttab <- sample_phosphodata
nameparam <- "testplot"

create_heatmap(counttab = counttab,
               colmetatable = NULL,
               colannotationlist = NULL, colclusterparam = FALSE,
               rowclusterparam = FALSE, nameparam)

```

---

deva

*Run the entire blacksheep Function from Start to finish*


---

### Description

Run the entire blacksheep Function from Start to finish

### Usage

```

deva(se, analyze_negative_outliers = FALSE,
     aggregate_features = FALSE, feature_delineator = "\\.",
     fraction_samples_cutoff = 0.3, fdrcutoffvalue = 0.1)

```

### Arguments

```

se                The SummarizedExperiment object containing the countdata and the associated
                  annotation data with comparisons in the colData object.

analyze_negative_outliers
                  DEFAULT: FALSE; Toggle the analysis of outliers in the negative direction as
                  well. Will lead to the output of the outlier table containing "-1" values, in addi-
                  tion to negative outputs for boundaries and aggregate tables (if applicable)

aggregate_features
                  DEFAULT: FALSE; Toggle the Aggregate feature, which will aggregate features
                  in your table based on the given delineator. Aggregation will output an aggregate
                  table that counts the number of outliers per feature, and also a fraction table that
                  show the number of outliers / number of candidates (which excludes missing
                  values)

feature_delineator
                  DEFAULT: <"\"> What character delineates the separation between primary
                  and secondary features. NOTE: to use proper R syntax with escape characters if
                  necessary Ex) Protein1.Phosphosite1 uses "\" to aggregate on Protein1

```

`fraction_samples_cutoff`  
 DEFAULT: 0.3; Input a fractional cut off for the of samples that need to have an outlier for feature to be considered. ex) 10 samples in ingroup - 3 need to have an outlier for feature to be considered significant

`fdr cutoffvalue` DEFAULT: 0.1; The FDR value for significance

### Value

outputs the full output of deva, including the analysis tables, the heatmaps for the analyses, the fraction table showing the fraction of outliers per sample, and the median and boundary values that together comprise the outlier boundary

### Examples

```
suppressPackageStartupMessages(library(SummarizedExperiment))
data("sample_phosphodata")
data("sample_annotationdata")

se <- SummarizedExperiment(
  assays = list(counts = as.matrix(sample_phosphodata[1:1000,])),
  colData = DataFrame(sample_annotationdata))

deva(se = se,
  analyze_negative_outliers = FALSE, aggregate_features = FALSE,
  feature_delineator = "-", fraction_samples_cutoff = 0.3,
  fdr cutoffvalue = 0.1)
```

---

<code>deva_normalization</code>	<i>Normalization of data to prepare for deva. Uses a Median of Ratio method followed by a log2 transformation.</i>
---------------------------------	--

---

### Description

Normalization of data to prepare for deva. Uses a Median of Ratio method followed by a log2 transformation.

### Usage

```
deva_normalization(intable, method = "MoR-log")
```

### Arguments

`intable` table with samples along the columns and features along the rows.

`method` DEFAULT: "MoR-log"; Method by which to normalize data in preparation for deva. Options are <"MoR-log", "MoR", "log">. Where "MoR" refers to the Median of ratio's. The "log" transformation is necessary to compress heavily skewed data and allow for proper detection. "MoR-log" as the default will perform MoR followed by a log2 transform.

**Value**

A normalized table for input into deva

**Examples**

```
library(pasilla)
pasCts <- system.file("extdata",
  "pasilla_gene_counts.tsv", package="pasilla")
cts <- as.matrix(read.csv(pasCts, sep="\t", row.names="gene_id"))
norm_cts <- deva_normalization(cts, method = "MoR-log")
```

---

deva\_results

*Utility function that allows easier grabbing of data*


---

**Description**

Utility function that allows easier grabbing of data

**Usage**

```
deva_results(deva_out, ID = NULL, type = NULL)
```

**Arguments**

deva_out	output from the deva function
ID	The keyword to search through analyses and grab desired output
type	<"table"   "heatmap"   "fraction_table"   "median"   "boundary"> to return the desired analysis type

**Value**

desired subset of analysis from deva

**Examples**

```
suppressPackageStartupMessages(library(SummarizedExperiment))
data("sample_phosphodata")
data("sample_annotationdata")

se = SummarizedExperiment(
  assays = list(counts = as.matrix(sample_phosphodata[1:1000,])),
  colData = DataFrame(sample_annotationdata))

deva_out = deva(se = se,
  analyze_negative_outliers = FALSE, aggregate_features = TRUE,
  feature_delineator = "-", fraction_samples_cutoff = 0.3,
  fdrcutoffvalue = 0.1)

deva_results(deva_out, ID = "outlieranalysis", type = "table")
```

---

```
make_comparison_columns
```

*Utility function that will take in columns with several subcategories, and output several columns each with binary classifications. ex) col1: A,B,C » colA: A,notA,notA; colB: notB,B,notB; colC: notC,notC,C*

---

### Description

Utility function that will take in columns with several subcategories, and output several columns each with binary classifications. ex) col1: A,B,C » colA: A,notA,notA; colB: notB,B,notB; colC: notC,notC,C

### Usage

```
make_comparison_columns(intable)
```

### Arguments

`intable`            table where each column has more than one subcategory, can be multiple columns

### Value

an expanded table with each of the columns as a binary labeling of each subcategory.

### Examples

```
data("sample_annotationdata")
new_comparisons <- make_comparison_columns(
  sample_annotationdata[,1,drop=FALSE])
```

---

```
make_outlier_table
```

*Separate out the "i"th gene, take the bounds, and then create a column that says whether or not this gene is high, low, or none in a sample with regards to the other samples in the dataset. Repeat this for every gene to create a reference table.*

---

### Description

Separate out the "i"th gene, take the bounds, and then create a column that says whether or not this gene is high, low, or none in a sample with regards to the other samples in the dataset. Repeat this for every gene to create a reference table.

### Usage

```
make_outlier_table(intable, analyze_negative_outliers = FALSE)
```



**Arguments**

`intable` table with all of the inputted information, samples along the x-axis, features along the y-axis

`analyze_negative_outliers` DEFAULT: FALSE; Toggle the analysis of outliers in the negative direction. Will lead to the output of the outlier table containing "-1" values, in addition to negative outputs for boundaries and aggregate tables (if applicable)

**Value**

a list with varied sections depending on parameters: `$outliertab` - table converted to outlier form with 0s, 1s, and -1s, `$upperboundtab` - list of upper boundaries for outliers `$lowerboundtab` - list of lower boundaries of outliers `$sampmedtab` - list of median value per feature

**Examples**

```
data("sample_phosphodata")
reftable_function_out <- make_outlier_table(sample_phosphodata[1:1000,],
  analyze_negative_outliers = FALSE)
outliertab <- reftable_function_out$outliertab
upperboundtab <- reftable_function_out$upperboundtab
lowerboundtab <- reftable_function_out$lowerboundtab
sampmedtab <- reftable_function_out$sampmedtab
```

---

<code>outlier_analysis</code>	<i>With the grouptablist generated by count_outliers - run through and run a fisher exact test to get the p.value for the difference in outlier count for each feature in each of your comparisons</i>
-------------------------------	--

---

**Description**

With the grouptablist generated by count\_outliers - run through and run a fisher exact test to get the p.value for the difference in outlier count for each feature in each of your comparisons

**Usage**

```
outlier_analysis(grouptablist, fraction_table = NULL,
  fraction_samples_cutoff = 0.3,
  write_out_tables = FALSE, outfilepath = tempdir())
```

**Arguments**

`grouptablist` table generated by the count\_outliers function. NOTE that the inputted grouptablist will be deciphered to determine its content. This means that user decides to input the outliertab or aggregate tab, and the output will analyze according to what positive and negative information is contained within the table

`fraction_table` DEFAULT: NULL; Input a fraction table to filter to only include features that have x an outlier.

`fraction_samples_cutoff` DEFAULT: 0.3; Input a fractional cut off for the of samples that need to have an outlier for feature to be considered. ex) 10 samples in ingroup - 3 need to have an outlier for feature to be considered significant

`write_out_tables`            DEFAULT: FALSE; utility in function to write out each of the analyses to a separate table to wherever `<outfilepath>` is specified.

`outfilepath`            the full string path to where the file should output to, DEFAULT is a `tempdir()`

**Value**

the analysis table with `p.value`, `fdr`, and raw data per comparison

**Examples**

```
data("sample_phosphodata")
reftable_function_out <- make_outlier_table(sample_phosphodata[1:1000,])
outliertab <- reftable_function_out$outliertab

data("sample_annotationdata")
groupings <- comparison_groupings(sample_annotationdata)

count_outliers_out <- count_outliers(groupings, outliertab,
  aggregate_features = FALSE)
grouptablist <- count_outliers_out$grouptablist
fractiontab <- count_outliers_out$fractiontab

outlier_analysis_out <- outlier_analysis(grouptablist,
  fraction_table = fractiontab)
```

---

<code>outlier_heatmap</code>	<i>With the grouptablist generated by <code>count_outliers</code> - run through and run a fisher exact test to get the <code>p.value</code> for the difference in outlier count for each feature in each of your comparisons</i>
------------------------------	--

---

**Description**

With the `grouptablist` generated by `count_outliers` - run through and run a fisher exact test to get the `p.value` for the difference in outlier count for each feature in each of your comparisons

**Usage**

```
outlier_heatmap(outlier_analysis_out, analysis_num = NULL,
  counttab, metatable, fdrcutoffvalue = 0.1)
```

**Arguments**

`outlier_analysis_out`            the full outlier\_analysis data objet

`analysis_num`            DEFAULT: NULL; if you only want to plot the heatmap for a particular analysis, enter number of that analysis

`counttab`            the raw data before outlier analysis

`metatable`            the complete metatable that was used to generate the comparisons, will be used for annotation of the heatmap

`fdrcutoffvalue`    DEFAULT: 0.1; The FDR value for significance

**Value**

outputs a pdf with the heatmap in the current working directory

**Examples**

```
data("sample_phosphodata")
reftable_function_out <- make_outlier_table(sample_phosphodata[1:1000,])
outliertab <- reftable_function_out$outliertab

data("sample_annotationdata")
groupings <- comparison_groupings(sample_annotationdata)

count_outliers_out <- count_outliers(groupings, outliertab,
  aggregate_features = FALSE)
grouptablist <- count_outliers_out$grouptablist
fractiontab <- count_outliers_out$fractiontab

outlier_analysis_out <- outlier_analysis(grouptablist,
  fraction_table = fractiontab)

metatable <- sample_annotationdata
counttab <- sample_phosphodata

hm1 <- outlier_heatmap(outlier_analysis_out, analysis_num = NULL,
  fractiontab, metatable, fdrcutoffvalue = 0.1)
```

---

sample\_annotationdata *sample\_annotationdata*

---

**Description**

Example annotation data for Outlier analysis. This example data is a subset of the data used in the CPTAC3 Breast Cancer exploration study: (doi: 10.1038/nature18003). Each row corresponds to a sample and each column is an binary annotation for that sample.

**Usage**

```
sample_annotationdata
```

**Format**

A data frame with 76 rows and 6 variables:

**PAM50\_Her2** The binary PAM50 Her2 classification for each sample  
**PAM50\_Basal** The binary PAM50 Basal classification for each sample  
**PAM50\_LumA** The binary PAM50 LumA classification for each sample  
**PAM50\_LumB** The binary PAM50 LumB classification for each sample  
**ER\_Status** The ER Status classification for each sample  
**PR\_Status** The PR Status classification for each sample ...

**Source**

<https://cptac-data-portal.georgetown.edu/cptac/s/S029>

---

sample_phosphodata	<i>sample_phosphodata</i>
--------------------	---------------------------

---

### Description

Example phosphoprotein data for Outlier analysis This example data is a subset of the data used in the CPTAC3 Breast Cancer exploration study: (doi: 10.1038/nature18003). Each row corresponds to a phosphoprotein site, and each column is a sample. The values within the table are normalized massspec phosphoprotein values.

### Usage

```
sample_phosphodata
```

### Format

A data frame with 15532 rows and 76 variables:

**TCGA-A2-A0CM** phosphoprotein levels for each gene  
**TCGA-A2-A0D2** phosphoprotein levels for each gene  
**TCGA-A2-A0EQ** phosphoprotein levels for each gene  
**TCGA-A2-A0EV** phosphoprotein levels for each gene  
**TCGA-A2-A0EX** phosphoprotein levels for each gene  
**TCGA-A2-A0EY** phosphoprotein levels for each gene  
**TCGA-A2-A0SW** phosphoprotein levels for each gene  
**TCGA-A2-A0SX** phosphoprotein levels for each gene  
**TCGA-A2-A0T3** phosphoprotein levels for each gene  
**TCGA-A2-A0T6** phosphoprotein levels for each gene  
**TCGA-A2-A0YC** phosphoprotein levels for each gene  
**TCGA-A2-A0YD** phosphoprotein levels for each gene  
**TCGA-A2-A0YF** phosphoprotein levels for each gene  
**TCGA-A2-A0YG** phosphoprotein levels for each gene  
**TCGA-A2-A0YM** phosphoprotein levels for each gene  
**TCGA-A7-A0CE** phosphoprotein levels for each gene  
**TCGA-A7-A0CJ** phosphoprotein levels for each gene  
**TCGA-A7-A13F** phosphoprotein levels for each gene  
**TCGA-A8-A06N** phosphoprotein levels for each gene  
**TCGA-A8-A06Z** phosphoprotein levels for each gene  
**TCGA-A8-A076** phosphoprotein levels for each gene  
**TCGA-A8-A079** phosphoprotein levels for each gene  
**TCGA-A8-A08Z** phosphoprotein levels for each gene  
**TCGA-A8-A09G** phosphoprotein levels for each gene  
**TCGA-AN-A04A** phosphoprotein levels for each gene  
**TCGA-AN-A0AJ** phosphoprotein levels for each gene

**TCGA-AN-A0AL** phosphoprotein levels for each gene  
**TCGA-AN-A0AM** phosphoprotein levels for each gene  
**TCGA-AN-A0FK** phosphoprotein levels for each gene  
**TCGA-AN-A0FL** phosphoprotein levels for each gene  
**TCGA-AO-A03O** phosphoprotein levels for each gene  
**TCGA-AO-A0J6** phosphoprotein levels for each gene  
**TCGA-AO-A0J9** phosphoprotein levels for each gene  
**TCGA-AO-A0JC** phosphoprotein levels for each gene  
**TCGA-AO-A0JE** phosphoprotein levels for each gene  
**TCGA-AO-A0JJ** phosphoprotein levels for each gene  
**TCGA-AO-A0JL** phosphoprotein levels for each gene  
**TCGA-AO-A0JM** phosphoprotein levels for each gene  
**TCGA-AO-A126** phosphoprotein levels for each gene  
**TCGA-AO-A12B** phosphoprotein levels for each gene  
**TCGA-AO-A12D** phosphoprotein levels for each gene  
**TCGA-AO-A12E** phosphoprotein levels for each gene  
**TCGA-AO-A12F** phosphoprotein levels for each gene  
**TCGA-AR-A0TR** phosphoprotein levels for each gene  
**TCGA-AR-A0TT** phosphoprotein levels for each gene  
**TCGA-AR-A0TV** phosphoprotein levels for each gene  
**TCGA-AR-A0TX** phosphoprotein levels for each gene  
**TCGA-AR-A0U4** phosphoprotein levels for each gene  
**TCGA-AR-A1AP** phosphoprotein levels for each gene  
**TCGA-AR-A1AS** phosphoprotein levels for each gene  
**TCGA-AR-A1AV** phosphoprotein levels for each gene  
**TCGA-AR-A1AW** phosphoprotein levels for each gene  
**TCGA-BH-A0AV** phosphoprotein levels for each gene  
**TCGA-BH-A0BV** phosphoprotein levels for each gene  
**TCGA-BH-A0C1** phosphoprotein levels for each gene  
**TCGA-BH-A0C7** phosphoprotein levels for each gene  
**TCGA-BH-A0DD** phosphoprotein levels for each gene  
**TCGA-BH-A0DG** phosphoprotein levels for each gene  
**TCGA-BH-A0E1** phosphoprotein levels for each gene  
**TCGA-BH-A0E9** phosphoprotein levels for each gene  
**TCGA-BH-A18N** phosphoprotein levels for each gene  
**TCGA-BH-A18Q** phosphoprotein levels for each gene  
**TCGA-BH-A18U** phosphoprotein levels for each gene  
**TCGA-C8-A12L** phosphoprotein levels for each gene  
**TCGA-C8-A12T** phosphoprotein levels for each gene  
**TCGA-C8-A12U** phosphoprotein levels for each gene

**TCGA-C8-A12V** phosphoprotein levels for each gene  
**TCGA-C8-A12Z** phosphoprotein levels for each gene  
**TCGA-C8-A130** phosphoprotein levels for each gene  
**TCGA-C8-A131** phosphoprotein levels for each gene  
**TCGA-C8-A134** phosphoprotein levels for each gene  
**TCGA-C8-A135** phosphoprotein levels for each gene  
**TCGA-C8-A138** phosphoprotein levels for each gene  
**TCGA-D8-A142** phosphoprotein levels for each gene  
**TCGA-E2-A154** phosphoprotein levels for each gene  
**TCGA-E2-A158** phosphoprotein levels for each gene

### Source

<https://cptac-data-portal.georgetown.edu/cptac/s/S029>

---

sample_rnadata	<i>sample_rnadata</i>
----------------	-----------------------

---

### Description

Example RNA data for Outlier analysis This example data is a subset of the data used in the CPTAC3 Breast Cancer exploration study: (doi: 10.1038/nature18003). Each row corresponds to a gene, and each column is a sample. The values within the table are normalized transcript counts.

### Usage

sample\_rnadata

### Format

A data frame with 4317 rows and 76 variables:

**TCGA-A2-A0CM** RNA levels for each gene  
**TCGA-A2-A0D2** RNA levels for each gene  
**TCGA-A2-A0EQ** RNA levels for each gene  
**TCGA-A2-A0EV** RNA levels for each gene  
**TCGA-A2-A0EX** RNA levels for each gene  
**TCGA-A2-A0EY** RNA levels for each gene  
**TCGA-A2-A0SW** RNA levels for each gene  
**TCGA-A2-A0SX** RNA levels for each gene  
**TCGA-A2-A0T3** RNA levels for each gene  
**TCGA-A2-A0T6** RNA levels for each gene  
**TCGA-A2-A0YC** RNA levels for each gene  
**TCGA-A2-A0YD** RNA levels for each gene  
**TCGA-A2-A0YF** RNA levels for each gene

**TCGA-A2-A0YG** RNA levels for each gene  
**TCGA-A2-A0YM** RNA levels for each gene  
**TCGA-A7-A0CE** RNA levels for each gene  
**TCGA-A7-A0CJ** RNA levels for each gene  
**TCGA-A7-A13F** RNA levels for each gene  
**TCGA-A8-A06N** RNA levels for each gene  
**TCGA-A8-A06Z** RNA levels for each gene  
**TCGA-A8-A076** RNA levels for each gene  
**TCGA-A8-A079** RNA levels for each gene  
**TCGA-A8-A08Z** RNA levels for each gene  
**TCGA-A8-A09G** RNA levels for each gene  
**TCGA-AN-A04A** RNA levels for each gene  
**TCGA-AN-A0AJ** RNA levels for each gene  
**TCGA-AN-A0AL** RNA levels for each gene  
**TCGA-AN-A0AM** RNA levels for each gene  
**TCGA-AN-A0FK** RNA levels for each gene  
**TCGA-AN-A0FL** RNA levels for each gene  
**TCGA-AO-A03O** RNA levels for each gene  
**TCGA-AO-A0J6** RNA levels for each gene  
**TCGA-AO-A0J9** RNA levels for each gene  
**TCGA-AO-A0JC** RNA levels for each gene  
**TCGA-AO-A0JE** RNA levels for each gene  
**TCGA-AO-A0JJ** RNA levels for each gene  
**TCGA-AO-A0JL** RNA levels for each gene  
**TCGA-AO-A0JM** RNA levels for each gene  
**TCGA-AO-A126** RNA levels for each gene  
**TCGA-AO-A12B** RNA levels for each gene  
**TCGA-AO-A12D** RNA levels for each gene  
**TCGA-AO-A12E** RNA levels for each gene  
**TCGA-AO-A12F** RNA levels for each gene  
**TCGA-AR-A0TR** RNA levels for each gene  
**TCGA-AR-A0TT** RNA levels for each gene  
**TCGA-AR-A0TV** RNA levels for each gene  
**TCGA-AR-A0TX** RNA levels for each gene  
**TCGA-AR-A0U4** RNA levels for each gene  
**TCGA-AR-A1AP** RNA levels for each gene  
**TCGA-AR-A1AS** RNA levels for each gene  
**TCGA-AR-A1AV** RNA levels for each gene  
**TCGA-AR-A1AW** RNA levels for each gene  
**TCGA-BH-A0AV** RNA levels for each gene

**TCGA-BH-A0BV** RNA levels for each gene  
**TCGA-BH-A0C1** RNA levels for each gene  
**TCGA-BH-A0C7** RNA levels for each gene  
**TCGA-BH-A0DD** RNA levels for each gene  
**TCGA-BH-A0DG** RNA levels for each gene  
**TCGA-BH-A0E1** RNA levels for each gene  
**TCGA-BH-A0E9** RNA levels for each gene  
**TCGA-BH-A18N** RNA levels for each gene  
**TCGA-BH-A18Q** RNA levels for each gene  
**TCGA-BH-A18U** RNA levels for each gene  
**TCGA-C8-A12L** RNA levels for each gene  
**TCGA-C8-A12T** RNA levels for each gene  
**TCGA-C8-A12U** RNA levels for each gene  
**TCGA-C8-A12V** RNA levels for each gene  
**TCGA-C8-A12Z** RNA levels for each gene  
**TCGA-C8-A130** RNA levels for each gene  
**TCGA-C8-A131** RNA levels for each gene  
**TCGA-C8-A134** RNA levels for each gene  
**TCGA-C8-A135** RNA levels for each gene  
**TCGA-C8-A138** RNA levels for each gene  
**TCGA-D8-A142** RNA levels for each gene  
**TCGA-E2-A154** RNA levels for each gene  
**TCGA-E2-A158** RNA levels for each gene

**Source**

<https://cptac-data-portal.georgetown.edu/cptac/s/S029>



# Index

## \* **blacksheepr**

- annotationlist\_builder, 2
- comparison\_groupings, 3
- count\_outliers, 3
- create\_heatmap, 4
- deva, 5
- deva\_normalization, 6
- deva\_results, 7
- make\_comparison\_columns, 8
- make\_outlier\_table, 8
- outlier\_analysis, 9
- outlier\_heatmap, 10

## \* **datasets**

- sample\_annotationdata, 11
- sample\_phosphodata, 12
- sample\_rnadata, 14

## \* **deva**

- annotationlist\_builder, 2
- comparison\_groupings, 3
- count\_outliers, 3
- create\_heatmap, 4
- deva, 5
- deva\_normalization, 6
- deva\_results, 7
- make\_comparison\_columns, 8
- make\_outlier\_table, 8
- outlier\_analysis, 9
- outlier\_heatmap, 10

## \* **outliers**

- annotationlist\_builder, 2
- comparison\_groupings, 3
- count\_outliers, 3
- create\_heatmap, 4
- deva, 5
- deva\_normalization, 6
- deva\_results, 7
- make\_comparison\_columns, 8
- make\_outlier\_table, 8
- outlier\_analysis, 9
- outlier\_heatmap, 10

annotationlist\_builder, 2

comparison\_groupings, 3

count\_outliers, 3  
create\_heatmap, 4

deva, 5  
deva\_normalization, 6  
deva\_results, 7

make\_comparison\_columns, 8  
make\_outlier\_table, 8

outlier\_analysis, 9  
outlier\_heatmap, 10

sample\_annotationdata, 11  
sample\_phosphodata, 12  
sample\_rnadata, 14