Package 'augere.de'

June 4, 2026

Version 0.99.2

Date 2026-04-11

Title Automatic Generation of Differential Expression Analyses

Description Implements pipelines for generating differential expression analysis reports in the augere framework.
This includes analyses with edgeR or voom-limma, with a variety of options for contrasts, blocking and covariates.
Each pipeline function generates a self-contained Rmarkdown report with all of the steps required to reproduce the DE analysis.

License MIT + file LICENSE

Imports utils, methods, limma, edgeR, S4Vectors, augere.core, alabaster.se

Suggests testthat, knitr, rmarkdown, BiocStyle, airway, SummarizedExperiment

VignetteBuilder knitr

RoxygenNote 7.3.3

Encoding UTF-8

URL https://github.com/augere-bioinfo/augere.de

BugReports https://github.com/augere-bioinfo/augere.de/issues

biocViews WorkflowManagement, ReportWriting, DifferentialExpression, Transcription

git_url https://git.bioconductor.org/packages/augere.de

git_branch devel

git_last_commit 9c39bb4

git_last_commit_date 2026-05-14

Repository Bioconductor 3.24

Date/Publication 2026-06-04

Author Aaron Lun [cre, aut] (ORCID: <https://orcid.org/0000-0002-3564-4813>)

Maintainer Aaron Lun <infinite.monkeys.with.keyboards@gmail.com>

Contents

`augere.de-package` *augere.de: Automatic Generation of Differential Expression Analyses*

Description

Implements pipelines for generating differential expression analysis reports in the augere framework. This includes analyses with edgeR or voom-limma, with a variety of options for contrasts, blocking and covariates. Each pipeline function generates a self-contained Rmarkdown report with all of the steps required to reproduce the DE analysis.

Author(s)

Maintainer: Aaron Lun <infinite.monkeys.with.keyboards@gmail.com> (ORCID)

See Also

Useful links:

- https://github.com/augere-bioinfo/augere.de
- Report bugs at https://github.com/augere-bioinfo/augere.de/issues

---

`findSubsetGroups` *Subset by groups in contrasts*

Description

List all groups involved in contrasts, to use with `subset.group=TRUE`. This is mostly intended for use by developers of differential analysis pipelines.

Usage

```
findSubsetGroups(contrast.info)
```

Arguments

`contrast.info`	List of contrast information as returned by `processSimpleComparisons`

Details

No subsetting by group is performed if covariate-based contrasts are present, as all samples are potentially informative in such an analysis.

Value

Character vector of the groups involved in `"versus"` or `"anova"` contrasts. If any covariate-based contrasts are present, `NULL` is always returned.

Author(s)

Aaron Lun

Examples

```
findSubsetGroups(processSimpleComparisons(c("disease", "healthy")))
findSubsetGroups(processSimpleComparisons(c("treated1", "treated2", "healthy")))
findSubsetGroups(processSimpleComparisons(c("age")))
```

---

loadExampleDataset *Load example dataset*

Description

Load an example RNA-seq dataset from the **airway** package. This is a convenience wrapper that strips out some unnecesssary metadata.

Usage

```
loadExampleDataset()
```

Value

A RangedSummarizedExperiment containing the **airway** dataset.

Author(s)

Aaron Lun

Examples

```
loadExampleDataset()
```

processContrastMetadata *Process contrast metadata*

Description

Create R commands to define the contrast metadata for the results of runEdgeR and runVoom.

Usage

```
processContrastMetadata(info)
```

Arguments

info	List containing information for a single contrast. This is typically an entry of the list returned by processSimpleComparisons or processCustomContrasts.

Value

Character vector containing arguments for constructing the gpsa.differential_gene_expression sublist of the result metadata.

Author(s)

Aaron Lun

Examples

```
contrast.info <- processSimpleComparisons(list(
    main=c("disease", "healthy"),
    secondary="dosage"
))
cat(processContrastMetadata(contrast.info[[1]]), sep="\n")
cat(processContrastMetadata(contrast.info[[2]]), sep="\n")
```

processCustomContrasts *Custom contrasts*

Description

Construct R commands to define custom contrast vectors/matrices.

Usage

```
processCustomContrasts(
  contrasts,
  design.name = "design",
  contrast.name = "con"
)
```

Arguments

contrasts	Numeric matrix or vector, a function to generate such a matrix/vector, or a character vector to be passed to makeContrasts, see Details. Alternatively, a list of such objects to specify multiple contrasts.
design.name	String containing the variable name of the design matrix.
contrast.name	String containing the variable name of the contrast vector/matrix.

Details

If contrasts is a a character vector, it will be passed to makeContrasts with levels= set to the design matrix. A vector of length 2 or more represents an ANOVA-like comparison.

If contrasts is a numeric vector, it should have names that match some or all of the column names of the design matrix. The values of this vector represent the entries of the contrast vector for the named coefficients; all other entries of the contrast vector are set to zero.

If contrasts is a numeric matrix, its row names should match some or all of the column names of the design matrix. The rows of this vector represent the rows of the contrast matrix for the named coefficients; all other entries of the contrast matrix are set to zero. The column names of the contrast matrix are set to those of contrasts, if available.

If contrasts is a function, it should accept the design matrix and return a contrast vector/matrrix. This mode is useful when the deparsed design matrix is difficult to read in the Rmarkdown report.

Value

A list containing one entry per comparison, where each entry contains:

- title, the title for the comparison. If comparisons is named, the corresponding (non-empty) name is used here, otherwise an appropriate title is automatically generated.
- type, the type of the comparison. This is always set to "custom".
- commands, character vector of R commands that produce the desired contrast vector/matrix. This assumes that the evaluation environment has a design matrix named design.name. The newly defined contrast vector/matrix will be stored as contrast.name in the environment.

Examples

```
processCustomContrasts(c(coef1 = 0.5, coef2 = 0.5, coef3 = -1))

mat <- matrix(0, 3, 2)
rownames(mat) <- c("grp1", "grp2", "grp3")
mat["grp1",] <- 1
mat["grp2",1] <- -1
mat["grp3",2] <- -1
processCustomContrasts(mat)

processCustomContrasts("TREATMENT - CONTROL")

processCustomContrasts(function(design) {
    out <- numeric(ncol(design))
    names(out) <- colnames(design)
    out[c("treated", "control")] <- c(1, -1)
    out
})
```

processCustomDesignMatrix *Create a custom design matrix*

Description

Construct R commands to create a custom design matrix, to be inserted into the generated Rmarkdown report.

Usage

```
processCustomDesignMatrix(design, se.name, design.name = "design")
```

Arguments

design	Function, formula or matrix specifying the experimental design.
se.name	String containing the variable name of the SummarizedExperiment object.
design.name	String containing the variable name of the design matrix.

Details

If design is a function, it should accept a SummarizedExperiment and return the matrix.

If design is a formula, it should use the column names of the SummarizedExperiment's colData. This is passed to model.matrix using data=colData(se) for a SummarizedExperiment se.

If design is a numeric matrix, it is deparsed and used verbatim. The number of rows of this matrix should be equal to the number of samples.

The design matrix is expected to be of full column rank.

Value

Character vector containing R commands that, upon evaluation, create a design matrix in the evaluation environment.

Author(s)

Aaron Lun

See Also

processSimpleDesignMatrix, for creating simple design matrices.

processCustomContrasts, to create custom contrasts.

Examples

```
cat(processCustomDesignMatrix(~ batch + treatment, "se"))

cat(processCustomDesignMatrix(function(x) {
    model.matrix(~ batch + treatment, data=colData(x))
}, "se"))

batch <- factor(rep(1:3, each=2))
```

```
treatment <- rep(c("Drg", "Ctrl"), 3)
mat <- model.matrix(~ batch + treatment)
cat(processCustomDesignMatrix(mat, "se"), sep="\n")
```

`processSimpleComparisons` *Contrasts for simple comparisons*

Description

Construct R commands to define contrast vectors/matrices for simple comparisons between groups or covariates.

Usage

```
processSimpleComparisons(
  comparisons,
  design.name = "design",
  contrast.name = "con"
)
```

Arguments

`comparisons`	Character vector specifying the groups to compare or the covariates to test. Non-character vectors will be coerced into character vectors. Alternatively, a (possibly named) list of such vectors where each entry represents a different comparison.
`design.name`	String containing the variable name of the design matrix.
`contrast.name`	String containing the variable name of the contrast vector/matrix.

Details

If a vector in `comparisons` is unnamed and of length 1, the sole entry is assumed to refer to a covariate in the `covariates` from `processSimpleDesignMatrix`. The null hypothesis is that the coefficient of the design matrix with the same name is zero, i.e., the covariate has no effect.

If a comparison vector is unnamed and of length 2, the entries represent the names of two groups in the specified `groups` factor. The null hypothesis is that there is no differential expression between the two groups. The log-fold change is defined as the first group (left) over the second (right).

If a comparison vector is unnamed and of length 3 or greater, the null hypothesis is that all of the specified levels of `groups` are equal. This is an ANOVA-like contrast where contrasts are formulated with respect to the last level, i.e., for `n` coefficients, `n-1` log-fold changes are reported representing the differences relative to the last coefficient.

If the comparison vector is named, all entries with the same name are assumed to represent a group of coefficients.

- If there is only one unique name, all entries of the vector are assumed to refer to entries of `covariates`. The null hypothesis is that the average of the coefficients for the specified covariates is equal to zero.

- If there are exactly two unique names, these are assumed to refer to two sets of entries of `groups`. The null hypothesis is that the average of the per-group coefficients are equal between the two sets.
- If there are three or more names, these are assumed to refer to multiple sets of entries of `groups`. The null hypothesis is that the average of the per-group coefficients are equal across all sets, i.e., an ANOVA-like comparison.

Value

A list containing one entry per comparison, where each entry contains:

- `title`, the title for the comparison. If `comparisons` is named, the corresponding (non-empty) name is used here, otherwise an appropriate title is automatically generated.
- `type`, the type of the comparison. This is one of `"covariate"` (for covariates), `"versus"` (for comparison between two groups or two sets of groups) or `"anova"` (for ANOVAs).
- `left` and `right`, lists of strings specifying the groups on the left (numerator) or right (denominator) of a "versus" comparison. Only present if `type = "versus"`.
- `groups`, list of lists of strings specifying the groups involved in an ANOVA-like comparison. Only present if `type = "anova"`.
- `covariate`, list of strings specifying the covariate(s) being tested. Only present if `type = "covariate"`.
- `commands`, character vector of R commands that produce the desired contrast vector/matrix. This assumes that the evaluation environment has a design matrix named `design.name`. The newly defined contrast vector/matrix will be stored as `contrast.name` in the environment.

See Also

`processSimpleDesignMatrix`, to generate the corresponding design matrix.

`processCustomContrasts`, to define more complex custom contrasts.

Examples

```
processSimpleComparisons(c("disease", "healthy"))
processSimpleComparisons("dosage")
processSimpleComparisons(c("untreated", "treated", "healthy"))
processSimpleComparisons(c(treated="treatment1", treated="treatment2", healthy="healthy"))
processSimpleComparisons(list(
    main=c("disease", "healthy"),
    secondary="dosage"
))
```

`processSimpleDesignMatrix` *Create a simple design matrix*

Description

Construct R commands to create a simple design matrix, to be inserted into the generated Rmarkdown report.

Usage

```
processSimpleDesignMatrix(
  groups,
  covariates,
  block,
  se.name,
  design.name = "design"
)
```

Arguments

groups	String specifying the colData column containing the grouping factor.
covariates	Character vector specifying the colData columns containing continuous covariates.
block	Character vector specifying colData columns containing additional uninteresting factrors.
se.name	String containing the variable name of the SummarizedExperiment object.
design.name	String containing the variable name of the design matrix.

Details

When creating the design matrix, groups, covariates and block are treated as additive factors.

- If groups is specified, the design matrix will not have an intercept. The first few columns will correspond to the levels of groups and are named by concatenating groups with the factor level. If groups is not supplied, the design matrix will have an intercept in the first column.
- If covariates are specified, they are represented by the columns after the per-group columns (if groups if supplied) or the intercept (otherwise).
- All remaining columns will correspond to the various levels of the block factors.

Syntactically invalid column names and levels can be used for all arguments.

The design matrix is expected to be of full column rank, e.g., groups is not confounded with elements of block.

Value

Character vector containing R commands that, upon evaluation, create a design matrix in the evaluation environment.

Author(s)

Aaron Lun

See Also

processCustomDesignMatrix, for creating more complex design matrices.

processSimpleComparisons, to create contrasts for this design matrix.

Examples

```
cat(processSimpleDesignMatrix("my_groups", "age", "batch", "se"), sep="\n")
```

reexports *Objects exported from other packages*

Description

These objects are imported from other packages. Follow the links below to see their documentation.

augere.core readResult, wrapInput

runEdgeR *Differential expression analysis with edgeR*

Description

Test for differentially expressed (DE) genes from an RNA-seq count matrix using the quasi-likelihood (QL) framework in **edgeR**.

Usage

```
runEdgeR(
  x,
  groups,
  comparisons,
  covariates = NULL,
  block = NULL,
  subset.factor = NULL,
  subset.levels = NULL,
  subset.groups = TRUE,
  design = NULL,
  contrasts = NULL,
  robust = TRUE,
  trend = TRUE,
  lfc.threshold = 0,
  assay = 1,
  row.data = NULL,
  metadata = NULL,
  output.dir = "edgeR",
  author = NULL,
  dry.run = FALSE,
  save.results = TRUE,
  suppress.plots = FALSE
)
```

Arguments

x	A SummarizedExperiment object containing a count matrix where genes and samples are in rows and columns, respectively. Alternatively, the output of wrapInput that refers to a SummarizedExperiment.

groups	String specifying the colData(x) column containing the grouping factor of interest, see processSimpleDesignMatrix for more details. This may be NULL for experimental designs with no groups, e.g., covariates only. Ignored if design and contrasts are provided.
comparisons	Character vector specifying two or more groups to compare from groups, or the covariate to be tested from covariates. Alternatively, or a list of such character vectors specifying multiple comparisons to perform. See processSimpleComparisons for more details. Ignored if design and contrasts are provided.
covariates	Character vector specifying the colData(x) columns containing continuous covariates of interest, see processSimpleDesignMatrix for more details. Ignored if design and contrasts are provided.
block	Character vector specifying the colData(x) columns containing additional (uninteresting) blocking factors, see processSimpleDesignMatrix for more details. Ignored if design and contrasts are provided.
subset.factor	String specifying the colData(x) column containing the factor to use for subsetting.
subset.levels	Vector containing the levels of the subset.factor to be retained.
subset.groups	Boolean indicating whether to automatically subset the dataset to only those samples assigned to groups in comparisons. Setting this to TRUE sacrifices some residual degrees of freedom for greater robustness against variability in irrelevant groups. Ignored if design and contrasts are provided. Also ignored if covariates is provided, as all samples are informative for a continuous covariate.
design	Matrix, function, or formula specifying the experimental design, see processCustomDesignMatrix for details. If this and contrasts are specified, groups, block, covariates, comparisons and subset.groups are ignored.
contrasts	String, function or matrix specifying a custom contrast, or a list of such objects; see processCustomContrasts for details. If this and design are specified, groups, block, covariates, comparisons and subset.groups are ignored.
robust	Boolean indicating whether robust empirical Bayes shrinkage should be used in glmQLFit. Setting this to TRUE sacrifices some precision for improved robustness against genes with extreme dispersions.
trend	Boolean indicating whether to shrink the QL dispersions towards a trend fitted to the mean in glmQLFit. Setting this to TRUE allows the analysis to adapt to difference mean-variance relationships, at the cost of some extra computational work.
lfc.threshold	Number specifying a log-fold change threshold for glmTreat. If zero, glmQLFTest is used instead. This should only be used for contrasts that can be formulated in terms of a single coefficient.
assay	String or integer specifying the assay of x containing the count matrix.
row.data	Character vector specifying the rowData(x) columns containing extra gene annotations to include in the result DataFrames.
metadata	Named list of additional metadata to store alongside each result.
output.dir	String containing the path to an output directory in which to write the Rmarkdown file and save results.
author	Character vector containg the names of the authors. If NULL, defaults to the current user.

dry.run	Boolean indicating whether to perform a dry run. This generates the Rmarkdown report in output.dir but does not execute the analysis.
save.results	Boolean indicating whether the results should be saved to file.
suppress.plots	Boolean indicating whether plots should be suppressed. This can be set to TRUE for faster execution.

Value

A Rmarkdown report named report.Rmd is written inside output.dir that contains the analysis commands.

If dry.run=FALSE, a list is returned containing:

- results, a list of DataFrames of tables from all contrasts. Each DataFrame corresponds to a comparison/contrast where each row corresponds to a gene (i.e., row) in se. Each DataFrame contains the following columns:
  - AveExpr, the average abundance.
  - F, the F-statistic.
  - LogFC, the log-fold change. (For non-ANOVA-like contrasts only.)
  - LogFC.<COLUMN>, the log-fold change corresponding to each column of the contrast matrix. (For ANOVA-like contrasts only.)
  - PValue, the p-value;
  - FDR, the Benjamini-Hochberg-adjusted p-value.
- normalized, a copy of x with normalized expression values. This contains:
  - lib.size and norm.factors columns in its colData, containing the library sizes and normalization factors, respectively.
  - a retained column in its rowData, indicating whether a gene was retained after filtering.
  - a logCPM assay, containing the log-counts-per-million after normalization.

  normalized may be subsetted by sample, depending on subset.factor, subset.group, etc.

If save.results=TRUE, the results are saved in a results directory inside output.

If dry.run=TRUE, NULL is returned. Only the Rmarkdown report is saved to file.

Author(s)

Aaron Lun

Examples

```
x <- loadExampleDataset()

tmp <- tempfile()
out <- runEdgeR(
    x,
    groups="dex",
    comparisons=c("trt", "untrt"),
    output=tmp
)

list.files(tmp, recursive=TRUE)
out
```

runVoom *Differential expression analysis with voom*

Description

Test for differentially expressed (DE) genes from an RNA-seq count matrix using the voom algorithm from **limma**.

Usage

```
runVoom(
  x,
  groups,
  comparisons,
  covariates = NULL,
  block = NULL,
  subset.factor = NULL,
  subset.levels = NULL,
  subset.groups = TRUE,
  design = NULL,
  contrasts = NULL,
  dc.block = NULL,
  robust = TRUE,
  trend = FALSE,
  quality = TRUE,
  lfc.threshold = 0,
  assay = 1,
  row.data = NULL,
  metadata = NULL,
  output.dir = "voom",
  author = NULL,
  dry.run = FALSE,
  save.results = TRUE,
  suppress.plots = FALSE
)
```

Arguments

x	A SummarizedExperiment object containing a count matrix where genes and samples are in rows and columns, respectively. Alternatively, the output of wrapInput that refers to a SummarizedExperiment.
groups	String specifying the colData(x) column containing the grouping factor of interest, see processSimpleDesignMatrix for more details. This may be NULL for experimental designs with no groups, e.g., covariates only. Ignored if design and contrasts are provided.
comparisons	Character vector specifying two or more groups to compare from groups, or the covariate to be tested from covariates. Alternatively, or a list of such character vectors specifying multiple comparisons to perform. See processSimpleComparisons for more details. Ignored if design and contrasts are provided.

covariates	Character vector specifying the colData(x) columns containing continuous covariates of interest, see processSimpleDesignMatrix for more details. Ignored if design and contrasts are provided.
block	Character vector specifying the colData(x) columns containing additional (uninteresting) blocking factors, see processSimpleDesignMatrix for more details. Ignored if design and contrasts are provided.
subset.factor	String specifying the colData(x) column containing the factor to use for subsetting.
subset.levels	Vector containing the levels of the subset.factor to be retained.
subset.groups	Boolean indicating whether to automatically subset the dataset to only those samples assigned to groups in comparisons. Setting this to TRUE sacrifices some residual degrees of freedom for greater robustness against variability in irrelevant groups. Ignored if design and contrasts are provided. Also ignored if covariates is provided, as all samples are informative for a continuous covariate.
design	Matrix, function, or formula specifying the experimental design, see processCustomDesignMatrix for details. If this and contrasts are specified, groups, block, covariates, comparisons and subset.groups are ignored.
contrasts	String, function or matrix specifying a custom contrast, or a list of such objects; see processCustomContrasts for details. If this and design are specified, groups, block, covariates, comparisons and subset.groups are ignored.
dc.block	String specifying the blocking factor to use in duplicateCorrelation. Typically used for uninteresting factors that cannot be used in block as they are confounded with the factors of interest. No additional blocking is performed if NULL.
robust	Boolean indicating whether robust empirical Bayes shrinkage should be used in eBayes. Setting this to TRUE sacrifices some precision for improved robustness against genes with extreme variances.
trend	Boolean indicating whether variances should be shrunk towards a trend in eBayes. Usually unnecessary as the observation weights already account for the mean-variance relationship.
quality	Boolean indicating whether quality weighting should be performed. This reduces the influence of low-quality samples at the cost of more computational work.
lfc.threshold	Number specifying a threshold on the log-fold change in treat. If zero, eBayes will be used instead.
assay	String or integer specifying the assay of x containing the count matrix.
row.data	Character vector specifying the rowData(x) columns containing extra gene annotations to include in the result DataFrames.
metadata	Named list of additional metadata to store alongside each result.
output.dir	String containing the path to an output directory in which to write the Rmarkdown file and save results.
author	Character vector containg the names of the authors. If NULL, defaults to the current user.
dry.run	Boolean indicating whether to perform a dry run. This generates the Rmarkdown report in output.dir but does not execute the analysis.
save.results	Boolean indicating whether the results should be saved to file.
suppress.plots	Boolean indicating whether plots should be suppressed. This can be set to TRUE for faster execution.

Value

A Rmarkdown report named `report.Rmd` is written inside `output.dir` that contains the analysis commands.

If `dry.run=FALSE`, a list is returned containing:

- `results`, a list of DataFrames of tables from all contrasts. Each DataFrame corresponds to a comparison/contrast where each row corresponds to a gene (i.e., row) in `se`. Each DataFrame contains the following columns:
  - `AveExpr`, the average abundance.
  - `t`, the F-statistic. (For non-ANOVA-like contrasts only.)
  - `F`, the F-statistic. (For ANOVA-like contrasts only.)
  - `LogFC`, the log-fold change. (For non-ANOVA-like contrasts only.)
  - `LogFC.<COLUMN>`, the log-fold change corresponding to each column of the contrast matrix. (For ANOVA-like contrasts only.)
  - `PValue`, the p-value;
  - `FDR`, the Benjamini-Hochberg-adjusted p-value.
- `normalized`, a copy of `x` with normalized expression values. This contains:
  - `lib.size` and `norm.factors` columns in its `colData`, containing the library sizes and normalization factors, respectively.
  - a `retained` column in its `rowData`, indicating whether a gene was retained after filtering.
  - a `logCPM` assay, containing the log-counts-per-million after normalization.

  `normalized` may be subsetted by sample, depending on `subset.factor`, `subset.group`, etc.

If `save.results=TRUE`, the results are saved in a `results` directory inside `output`.

If `dry.run=TRUE`, `NULL` is returned. Only the Rmarkdown report is saved to file.

Author(s)

Aaron Lun

Examples

```
x <- loadExampleDataset()

tmp <- tempfile()
out <- runVoom(
    x,
    groups="dex",
    comparisons=c("trt", "untrt"),
    output=tmp
)

list.files(tmp, recursive=TRUE)
out
```

Index