

# Package ‘kimma’

July 23, 2021

**Type** Package

**Title** Kinship In Mixed Model Analysis of RNA-seq

**Version** 0.1.0

**Author** Kim Dill-McFarland

**Maintainer** Kim Dill-McFarland <kadm@uw.edu>

**Description** Data analysis and linear mixed effects models with pairwise kinship for RNA-seq data.

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**biocViews**

**Imports** broom, car, coxme, data.table, doParallel, dplyr, edgeR, emmeans, forcats, foreach, ggplot2, limma, lme4, magrittr, readr, stats, tibble, tidyr, tidyselect, WGCNA

**RoxygenNote** 7.1.1

**Depends** R (>= 2.10)

## R topics documented:

align_metrics . . . . .	2
dat.example . . . . .	2
dat.voom.example . . . . .	3
extract_lmFit . . . . .	5
filter_rare . . . . .	6
kin.example . . . . .	6
kmFit . . . . .	7
make_modules . . . . .	9
summarise_kmFit . . . . .	10
summarise_lmFit . . . . .	11
<b>Index</b>	<b>12</b>

---

align_metrics	<i>Extract and format cleaning and alignment metrics</i>
---------------	----------------------------------------------------------

---

### Description

Extract data from FastQC trim settings, Picard, samtools flagstat, and featureCounts output by the RNA-seq fastq pipeline

### Usage

```
align_metrics(
  data.dir = NULL,
  trim = TRUE,
  bam = TRUE,
  picard = TRUE,
  bam.filter = TRUE,
  count = TRUE
)
```

### Arguments

data.dir	Character string of directory containing all associated files
trim	Logical if should include FastQC trim settings
bam	Logical if should include samtools flagstat for raw alignments
picard	Logical if should include Picard for raw alignments
bam.filter	Logical if should include samtools flagstat for filtered alignments
count	Logical if should include featureCounts total reads in genes

### Value

Data frame cleaning and alignment metrics for all libraries

---

dat.example	<i>kimma example DGEList.</i>
-------------	-------------------------------

---

### Description

An edgeR DGEList data set containing unnormalized RNA-seq counts. RNA-seq of human dendritic cells cultured with and without virus. Samples from 3 donors and a random subset of 1000 genes were selected. Counts are unnormalized.

### Usage

```
dat.example
```

## Format

Formal class 'DGEList' [package "edgeR"] with 1 slot:

1. **counts** A matrix with 1000 rows and 6 columns  
**rownames** character. ENSEMBL gene ID.  
**lib1** integer. Counts in library 1.  
**lib2** integer. Counts in library 2.  
**lib3** integer. Counts in library 3.  
**lib4** integer. Counts in library 4.  
**lib5** integer. Counts in library 5.  
**lib6** integer. Counts in library 6.
2. **samples** A data frame with 6 rows and 7 columns  
**group** factor. No grouping was provided. All = 1.  
**lib.size** numeric. Total library size for this 1000 gene subset.  
**norm.factors** numeric. Normalization factors. No normalization was completed. All = 1.  
**libID** character. Unique library ID. Matches column names in counts.  
**donorID** character. Donor ID.  
**median\_cv\_coverage** numeric. Median coefficient of variation of coverage. Quality metric for sequencing libraries calculated from original full data set.  
**virus** character. A for media samples with no virus. B for virus-infected samples.
3. **genes** A data frame with 1000 rows and 5 columns  
**hgnc\_symbol** character. Current approved HGNC symbol.  
**Previous symbols** character. Previous HGNC symbols.  
**Alias symbols** character. Alias HGNC symbols.  
**gene\_biotype** character. Gene product type. All = protein-coding.  
**geneName** character. ENSEMBL gene ID. Matches row names in counts.

## Source

[https://github.com/altman-lab/P259\\_pDC\\_public](https://github.com/altman-lab/P259_pDC_public)

## References

Dill-McFarland et al. 2021. Eosinophil-mediated suppression and Anti-IL-5 enhancement of plasmacytoid dendritic cell interferon responses in asthma. J Allergy Clin Immunol. In revision

---

dat.voom.example	<i>limma example EList.</i>
------------------	-----------------------------

---

## Description

A limma EList data set containing normalized log2 RNA-seq counts. RNA-seq of human dendritic cells cultured with and without virus. Samples from 3 donors and a random subset of 1000 genes were selected. Counts are TMM normalized log2 counts per million (CPM).

## Usage

dat.voom.example

## Format

Formal class 'EList' [package "limma"] with 1 slot:

1. **genes** A data frame with 1000 rows and 5 columns
  - hgnc\_symbol** character. Current approved HGNC symbol.
  - Previous symbols** character. Previous HGNC symbols.
  - Alias symbols** character. Alias HGNC symbols.
  - gene\_biotype** character. Gene product type. All = protein-coding.
  - geneName** character. ENSEMBL gene ID. Matches row names in E.
2. **targets** A data frame with 6 rows and 7 columns
  - group** factor. No grouping was provided. All = 1.
  - lib.size** numeric. Total library size for this 1000 gene subset.
  - norm.factors** numeric. TMM normalization factors.
  - libID** character. Unique library ID. Matches column names in E.
  - donorID** character. Donor ID.
  - median\_cv\_coverage** numeric. Median coefficient of variation of coverage. Quality metric for sequencing libraries calculated from original full data set.
  - virus** character. A for media samples with no virus. B for virus-infected samples.
3. **E** A matrix with 1000 rows and 6 columns
  - rownames** character. ENSEMBL gene ID.
  - lib1** integer. log2 CPM in library 1.
  - lib2** integer. log2 CPM in library 2.
  - lib3** integer. log2 CPM in library 3.
  - lib4** integer. log2 CPM in library 4.
  - lib5** integer. log2 CPM in library 5.
  - lib6** integer. log2 CPM in library 6.
4. **weights** A matrix with 1000 rows and 6 columns
  - 1** numeric. limma gene weights for library 1.
  - 2** numeric. limma gene weights for library 2.
  - 3** numeric. limma gene weights for library 3.
  - 4** numeric. limma gene weights for library 4.
  - 5** numeric. limma gene weights for library 5.
  - 6** numeric. limma gene weights for library 6.
5. **design** A matrix with 6 rows and 1 column
  - GrandMean** numeric. limma default design matrix.

## Source

[https://github.com/altman-lab/P259\\_pDC\\_public](https://github.com/altman-lab/P259_pDC_public)

## References

Dill-McFarland et al. 2021. Eosinophil-mediated suppression and Anti-IL-5 enhancement of plasmacytoid dendritic cell interferon responses in asthma. J Allergy Clin Immunol. In revision

---

extract_lmFit	<i>Extract lmFit model results</i>
---------------	------------------------------------

---

## Description

Extract model fit and significance for all individual variables and/or contrasts in a limma model

## Usage

```
extract_lmFit(
  design,
  fit,
  contrast.mat = NULL,
  dat.genes = NULL,
  name.genes = "geneName"
)
```

## Arguments

design	model matrix output by <code>model.matrix()</code>
fit	MArrayLM model fit output by <code>limma::eBayes()</code>
contrast.mat	contrast matrix output by <code>limma::makeContrasts()</code>
dat.genes	data frame with additional gene annotations. Optional.
name.genes	character for variable name in <code>dat.genes</code> that matches gene names in fit

## Value

Data frame with model fit and significance for all variable and genes. Format as in `limma::topTable()`

## Examples

```
# Run limma model
design <- model.matrix(~ virus, data = dat.voom.example$targets)
fit <- limma::eBayes(limma::lmFit(dat.voom.example$E, design))

## Get results
result <- extract_lmFit(design = design, fit = fit)
## Get results and add gene annotations
fdr <- extract_lmFit(design = design, fit = fit,
  dat.genes = dat.voom.example$genes, name.genes = "geneName")

# Run limma contrasts model
design <- model.matrix(~ 0 + virus, data = dat.voom.example$targets)
fit <- limma::lmFit(dat.voom.example$E, design)
contrast.mat <- limma::makeContrasts(virusB-virusA, levels = design)
fit <- eBayes(contrasts.fit(fit, contrast.mat))

## Get contrast results
fdr <- extract_lmFit(design = design, fit = fit, contrast.mat = contrast.mat)
```

---

filter_rare	<i>Filter rare and low abundance genes</i>
-------------	--------------------------------------------

---

### Description

Filter genes at a minimum counts per million (CPM) in a minimum number or percent of total samples.

### Usage

```
filter_rare(
  dat,
  min.CPM,
  gene.var = "geneName",
  min.sample = NULL,
  min.pct = NULL,
  plot = FALSE
)
```

### Arguments

dat	DGEList output by edgeR::DEGList()
min.CPM	numeric minimum counts per million (CPM)
gene.var	character name for column with gene names in dat\$genes that matches names in expression data dat\$E. Default "geneName"
min.sample	numeric minimum number of samples
min.pct	numeric minimum percent of samples (0-100)
plot	logical if should plot mean variance trends

### Value

DGEList object filtered to not rare genes

### Examples

```
dat.filter <- filter_rare(dat = dat.example, min.CPM = 0.1, min.sample = 3)
dat.filter <- filter_rare(dat = dat.example, min.CPM = 0.1, min.pct = 10, plot = TRUE)
```

---

kin.example	<i>kinma example kinship.</i>
-------------	-------------------------------

---

### Description

Matrix of pairwise kinship values between donor 1,2,3. Values are dummy data with 1 for self comparison, 0.5 for siblings, and 0.1 for unrelated.

### Usage

```
kin.example
```

**Format**

A matrix with 3 rows and 3 variables:

**rowname** Donor ID. Same as column names

**donor1** numeric kinship (0-1) with donor 1

**donor2** numeric kinship (0-1) with donor 2

**donor3** numeric kinship (0-1) with donor 3

---

kmFit

---

*Linear mixed effects models with kinship for RNA-seq*


---

**Description**

Run lmeKin and corresponding lm or lme without kinship of gene expression in RNA-seq data

**Usage**

```
kmFit(
  dat = NULL,
  kin = NULL,
  patientID = "ptID",
  libraryID = "libID",
  counts = NULL,
  meta = NULL,
  genes = NULL,
  subset.var = NULL,
  subset.lvl = NULL,
  subset.genes = NULL,
  model,
  compare.lm = FALSE,
  compare.lme = FALSE,
  contrast = FALSE,
  contrast.mat = NULL,
  processors = 1,
  p.method = "BH"
)
```

**Arguments**

dat	EList object output by voom( ). Contains counts (dat\$E), meta (dat\$targets), and genes (dat\$genes).
kin	Matrix with pairwise kinship values between individuals. Must be numeric with rownames.
patientID	Character of variable name to match dat\$targets to kinship row and column names.
libraryID	Character of variable name to match dat\$targets to dat\$E colnames
counts	Matrix of normalized expression. Rows are genes, columns are libraries.
meta	Matrix or data frame of sample and individual metadata.

genes	Matrix or data frame of gene metadata.
subset.var	Character list of variable name(s) to filter data by.
subset.lvl	Character list of variable value(s) or level(s) to filter data to. Must match order of subset.var
subset.genes	Character vector of genes to include in models.
model	Character vector of model starting with ~ Should include (1 patientID) if mixed effects will be run
compare.lm	Logical if should run corresponding lm model without kinship
compare.lme	Logical if should run corresponding lme model without kinship
contrast	Logical if should run pairwise contrasts. If no matrix provided, all possible pairwise comparisons are completed.
contrast.mat	Numeric contrast matrix created limma::makeContrasts( )
processors	Numeric processor to run in parallel
p.method	Character of FDR adjustment method. Values as in p.adjust( )

## Value

Dataframe with model fit and significance for each gene

## Examples

```
# All samples and all genes
# Not run
# kmFit(dat = dat.voom.example,
#       patientID = "donorID", libraryID = "libID",
#       kin = kin.example, compare.lme = TRUE,
#       model = "~ virus + (1|donorID)")

# Subset samples and genes
kmFit(dat = dat.voom.example,
      patientID = "donorID", libraryID = "libID",
      kin = kin.example,
      compare.lme = TRUE,
      subset.var = list("donorID"), subset.lvl = list(c("donor1", "donor2")),
      subset.genes = c("ENSG00000250479", "ENSG00000250510", "ENSG00000255823"),
      model = "~ virus + (1|donorID)")

# Pairwise contrasts
kmFit(dat = dat.voom.example,
      patientID = "donorID", libraryID = "libID",
      kin = kin.example,
      compare.lm = TRUE, contrast = TRUE,
      subset.genes = c("ENSG00000250479", "ENSG00000250510", "ENSG00000255823"),
      model = "~ donorID + (1|donorID)")
```



---

make\_modules

Construct WGCNA modules and associated data

---

## Description

Make WGCNA modules from gene expression data with dynamic soft threshold selection. Also outputs mean module expression and DAVID formatted gene lists

## Usage

```
make_modules(
  dat,
  genes = NULL,
  Rsq.min = NULL,
  sft.value = NULL,
  minModuleSize = 20,
  deepSplit = 3,
  nThread = 2
)
```

## Arguments

dat	limma EList output by voom( )
genes	Character vector of genes to used in module building. Must match rownames in dat. If not set, all genes in dat are used
Rsq.min	Numeric minimum R-squared for soft threshold selection. If set, sft.value is not used
sft.value	Numeric soft threshold. Set when minimum R-squared is no used
minModuleSize	Numeric minimum module size
deepSplit	Integer value between 0 and 4. Provides a simplified control over how sensitive module detection should be to module splitting, with 0 least and 4 most sensitive
nThread	Integer for number of threads to use

## Value

List including:

- genes Character vector of genes used in module building
- sft Data frame with soft thresholding selected for module building. Includes power, minimum R-squared, and connectivity
- sft.plot ggplot object of soft thresholding topology and connectivity
- mods Data frame of genes in modules
- mods.voom Data frame of mean module expression in each library
- david DAVID formatted data frame of genes in modules

## Examples

```
dat.mods <- make_modules(dat = dat.voom.example, sft.value = 1)
```

---

summarise_kmFit	<i>Summarise kmFit FDR results</i>
-----------------	------------------------------------

---

## Description

Summarise number of significant genes at various FDR cutoffs. Can split by up/down fold change as well.

## Usage

```
summarise_kmFit(
  fdr,
  fdr.cutoff = c(0.05, 0.1, 0.2, 0.3, 0.4, 0.5),
  contrast = FALSE,
  FCgroup = FALSE,
  intercept = FALSE
)
```

## Arguments

fdr	data.frame output by kimma::kmFit()
fdr.cutoff	numeric vector of FDR cutoffs to summarise at
contrast	logical if should separate summary by pairwise contrasts within variables
FCgroup	logical if should separate summary by up/down fold change groups
intercept	logical if should include intercept variable in summary

## Value

Data frame with total significant genes for each variable at various FDR cutoffs

## Examples

```
# Run kimma model
fdr <- kmFit(dat = dat.voom.example,
  patientID = "donorID", libraryID = "libID",
  kin = kin.example,
  compare.lme = TRUE,
  subset.genes = c("ENSG00000250479", "ENSG00000250510", "ENSG00000255823"),
  model = "~ virus + (1|donorID)")

# Summarise results
fdr.summary <- summarise_kmFit(fdr = fdr, fdr.cutoff = c(0.05, 0.5), FCgroup = TRUE)
```

---

summarise_lmFit	<i>Summarise lmFit FDR results</i>
-----------------	------------------------------------

---

**Description**

Summarise number of significant genes at various FDR cutoffs. Can split by up/down fold change as well.

**Usage**

```
summarise_lmFit(  
  fdr,  
  fdr.cutoff = c(0.05, 0.1, 0.2, 0.3, 0.4, 0.5),  
  FCgroup = FALSE,  
  intercept = FALSE  
)
```

**Arguments**

fdr	data.frame output by kimma::extract_lmFit( )
fdr.cutoff	numeric vector of FDR cutoffs to summarise at
FCgroup	logical if should separate summary by up/down fold change groups
intercept	logical if should include intercept variable in summary

**Value**

Data frame with total significant genes for each variable at various FDR cutoffs

**Examples**

```
# Run limma model  
design <- model.matrix(~ virus, data = dat.voom.example$targets)  
fit <- limma::eBayes(limma::lmFit(dat.voom.example$E, design))  
  
## Get results  
fdr <- extract_lmFit(design = design, fit = fit)  
  
# Summarise results  
fdr.summary <- summarise_lmFit(fdr = fdr, fdr.cutoff = c(0.05, 0.5), FCgroup = TRUE)
```

# Index

- \* **datasets**
  - dat.example, [2](#)
  - dat.voom.example, [3](#)
  - kin.example, [6](#)
- align\_metrics, [2](#)
- dat.example, [2](#)
- dat.voom.example, [3](#)
- extract\_lmFit, [5](#)
- filter\_rare, [6](#)
- kin.example, [6](#)
- kmFit, [7](#)
- make\_modules, [9](#)
- summarise\_kmFit, [10](#)
- summarise\_lmFit, [11](#)