Package 'RESOLVE'

May 26, 2026

Version 1.15.0

Date 2026-03-09

Title RESOLVE: An R package for the efficient analysis of mutational signatures from cancer genomes

Depends R (>= 4.1.0)

Imports Biostrings, BSgenome, BSgenome.Hsapiens.1000genomes.hs37d5, cluster, data.table, GenomeInfoDb, GenomicRanges, glmnet, ggplot2, gridExtra, IRanges, lsa, MutationalPatterns, nnls, parallel, reshape2, S4Vectors, RhpcBLASctl, survival

Suggests BiocGenerics, BiocStyle, testthat, knitr

Description Cancer is a genetic disease caused by somatic mutations in genes controlling key biological functions such as cellular growth and division. Such mutations may arise both through cell-intrinsic and exogenous processes, generating characteristic mutational patterns over the genome named mutational signatures. The study of mutational signatures have become a standard component of modern genomics studies, since it can reveal which (environmental and endogenous) mutagenic processes are active in a tumor, and may highlight markers for therapeutic response. Mutational signatures computational analysis presents many pitfalls. First, the task of determining the number of signatures is very complex and depends on heuristics. Second, several signatures have no clear etiology, casting doubt on them being computational artifacts rather than due to mutagenic processes. Last, approaches for signatures assignment are greatly influenced by the set of signatures used for the analysis. To overcome these limitations, we developed RESOLVE (Robust EStimation Of mutationaL signatures Via rEgularization), a framework that allows the efficient extraction and assignment of mutational signatures. RESOLVE implements a novel algorithm that enables (i) the efficient extraction, (ii) exposure estimation, and (iii) confidence assessment during the computational inference of mutational signatures.

Encoding UTF-8

License file LICENSE

URL https://github.com/danro9685/RESOLVE

BugReports https://github.com/danro9685/RESOLVE/issues

biocViews BiomedicalInformatics, SomaticMutation

RoxygenNote 7.3.3

VignetteBuilder knitr

git_url https://git.bioconductor.org/packages/RESOLVE

git_branch devel

git_last_commit 71166a9

git_last_commit_date 2026-04-28

Repository Bioconductor 3.24

Date/Publication 2026-05-25

Author Daniele Ramazzotti [aut] (ORCID:
<https://orcid.org/0000-0002-6087-2666>),
Luca De Sano [cre, aut] (ORCID:
<https://orcid.org/0000-0002-9618-3774>)

Maintainer Luca De Sano <luca.desano@gmail.com>

Contents

`associationAlterations` *associationAlterations*

Description

Perform the estimation of the mutations associated to specific mutational signatures.

Usage

```
associationAlterations(alterations, signatures)
```

Arguments

`alterations`	Matrix with the observed mutations (columns) for each patient (rows) as binary variables.
`signatures`	Matrix with the estimated exposures to signatures (columns) for each patient (rows).

Value

A list with the association estimates. It includes 5 elements: alterations: matrix with the observed mutations (columns) for each patient (rows) as binary variables. signatures: matrix with the estimated exposures to signatures (columns) for each patient (rows). associations: estimated associations. intercept: intercept resulting from the regularized regression estimate. fold_changes: fold changes estimating the strength of the associations.

Examples

```
data(association_mutations)
set.seed(12345)
alterations = association_mutations$alterations
normalized_alpha = association_mutations$normalized_alpha
association_alterations = associationAlterations(alterations = alterations,
  signatures = normalized_alpha)
```

`associationPrognosis` *associationPrognosis*

Description

Perform the estimation of mutational signatures associated to prognosis.

Usage

```
associationPrognosis(clinical_data, signatures)
```

Arguments

clinical_data	Matrix with two columns proving survival times (first column) and status (second column).
signatures	Matrix with the estimated exposures to signatures (columns) for each patient (rows).

Value

A vector of coefficients resulting from regularized Cox regression.

Examples

```
data(association_survival)
set.seed(12345)
clinical_data = association_survival$clinical_data
normalized_alpha = association_survival$normalized_alpha
prognosis_associations = associationPrognosis(clinical_data = clinical_data,
  signatures = normalized_alpha)
```

---

associationSignatures *associationSignatures*

Description

Perform the estimation of the mutational signatures associated to mutations in specific genes.

Usage

```
associationSignatures(alterations, signatures)
```

Arguments

alterations	Matrix with the observed mutations (columns) for each patient (rows) as binary variables.
signatures	Matrix with the estimated exposures to signatures (columns) for each patient (rows).

Value

A list with the association estimates. It includes 6 elements: alterations: matrix with the observed mutations (columns) for each patient (rows) as binary variables. signatures: matrix with the estimated exposures to signatures (columns) for each patient (rows). associations: estimated associations. intercept: intercept resulting from the regularized regression estimate. probabilities: probability of observing a mutation in a certain gene given the presence of specific signatures. fold_changes: fold changes estimating the strength of the associations.

Examples

```
data(association_mutations)
set.seed(12345)
alterations = association_mutations$alterations
normalized_alpha = association_mutations$normalized_alpha
association_signatures = associationSignatures(alterations = alterations,
  signatures = normalized_alpha)
```

---

association_mutations *List providing mutations as binary variables and signatures assignments for 656 breast cancer patients*

Description

Mutations and SBS assignments for 656 breast tumors from ICGC.

Usage

```
data(association_mutations)
```

Format

List providing mutations as binary variables and signatures assignments for 656 breast cancer patients from ICGC (https://dcc.icgc.org/)

Value

Mutations and SBS assignments

Source

ICGC data portal (https://dcc.icgc.org/).

---

association_survival *List providing clinical data and signatures assignments for 359 pancreatic cancer patients*

Description

Clinical data and SBS assignments for 359 pancreatic tumors from ICGC.

Usage

```
data(association_survival)
```

Format

List providing clinical data and signatures assignments for 359 pancreatic cancer patients from ICGC (https://dcc.icgc.org/)

Value

Clinical data and SBS assignments

Source

ICGC data portal (https://dcc.icgc.org/).

---

background *Germline replication error*

Description

Germline replication error estimated in Rahbari, Raheleh, et al. (2016).

Usage

```
data(background)
```

Format

Vector of rates

Value

Vector of rates for the 96 trinucleotides

Source

Nat Genet. 2016 Feb;48(2):126-133 (https://www.nature.com/articles/ng.3469).

---

background2 *COSMIC replication error*

Description

Background replication error signature derived from COSMIC SBS5.

Usage

```
data(background2)
```

Format

Vector of rates

Value

Vector of rates for the 96 trinucleotides

Source

COSMIC database (https://cancer.sanger.ac.uk/cosmic/signatures) v3.

cn_example_reduced *A reduced version of the copy number data for 5 TCGA samples in the format compatible with the import function*

Description

Reduced version of the dataset of counts of copy numbers detected in TCGA tumors published in Steele, Christopher D., et al. (2022).

Usage

```
data(cn_example_reduced)
```

Format

Reduced version of the counts of copy numbers in the format compatible with the import function

Value

Reduced version of the counts of copy numbers for 5 tumors and 48 copy number classes in the format compatible with the import function

Source

Nature. 2022 Jun;606(7916):984-991 (https://www.nature.com/articles/s41586-022-04738-6).

getCNCounts *getCNCounts*

Description

Create Copy Numbers (CNs) counts matrix from input data. This function has been derived from: https://github.com/UCL-Research-Department-of-Pathology/panConusig/blob/main/R/setup_CNsigs.R

Usage

```
getCNCounts(data)
```

Arguments

data	A data.frame with copy number data having 6 columns: sample name, chromosome, start position, end position, major CN, minor CN.

Value

A matrix with Copy Numbers (CNs) counts per patient.

Examples

```
data(cn_example_reduced)
res <- getCNCounts(data = cn_example_reduced)
```

getIDCounts *getIDCounts*

Description

Create Small Insertions and Deletions (IDs) counts matrix from input data.

Usage

```
getIDCounts(data, reference = NULL)
```

Arguments

data	A data.frame with variants having 6 columns: sample name, chromosome, start position, end position, ref, alt.
reference	A BSgenome object with the reference genome to be used.

Value

A matrix with Small Insertions and Deletions (IDs) counts per patient.

Examples

```
library('BSgenome.Hsapiens.1000genomes.hs37d5')
data(id_example_reduced)
res <- getIDCounts(data = id_example_reduced, reference = BSgenome.Hsapiens.1000genomes.hs37d5)
```

getMNVCounts *getMNVCounts*

Description

Create Multi-Nucleotide Variants (MNVs) counts matrix from input data.

Usage

```
getMNVCounts(data, predefined_dbs_mbs = FALSE)
```

Arguments

data	A data.frame with variants having 6 columns: sample name, chromosome, start position, end position, ref, alt.
predefined_dbs_mbs	Boolean. As defined by the function get_mut_type from the package MutationalPatterns, it specifies whether dbs and mbs mutations have been predefined in the input data. This function by default assumes that dbs and mbs mutations are present in the vcf as snvs, which are positioned next to each other. If your dbs/mbs mutations are called separately, you should set this argument to TRUE.

Value

A matrix with Multi-Nucleotide Variants (MNVs) counts per patient.

Examples

```
data(ssm560_reduced)
res <- getMNVCounts(data = ssm560_reduced)
```

`getSBSCounts` *getSBSCounts*

Description

Create Single Base Substitutions (SBS) counts matrix from input data for a provided reference genome.

Usage

```
getSBSCounts(data, reference = NULL)
```

Arguments

`data`	A data.frame with variants having 6 columns: sample name, chromosome, start position, end position, ref, alt.
`reference`	A BSgenome object with the reference genome to be used to retrieve flanking bases.

Value

A matrix with Single Base Substitutions (SBS) counts per patient.

Examples

```
library('BSgenome.Hsapiens.1000genomes.hs37d5')
data(ssm560_reduced)
res <- getSBSCounts(data = ssm560_reduced, reference = BSgenome.Hsapiens.1000genomes.hs37d5)
```

groupsCNPlot *groupsCNPlot*

Description

Plot observed Copy Number (CN) counts for different groups of patients.

Usage

```
groupsCNPlot(counts, groups, normalize = TRUE, xlabels = FALSE)
```

Arguments

counts	Matrix with Copy Number (CN) counts data.
groups	List where names are groups labels and elements are patients labels corresponding to rownames in counts.
normalize	Boolean value; shall I normalize observed counts?
xlabels	Boolean value; shall I display x labels?

Value

A ggplot2 object.

Examples

```
data(plot_data_examples)
counts <- plot_data_examples[['groups.CN.plot']][['counts']]
groups <- plot_data_examples[['groups.CN.plot']][['groups']]
groupsCNPlot(counts=counts,groups=groups)
```

groupsCXPlot *groupsCXPlot*

Description

Plot observed Copy Number (Reduced, CX) counts for different groups of patients.

Usage

```
groupsCXPlot(counts, groups, normalize = TRUE, xlabels = FALSE)
```

Arguments

counts	Matrix with Copy Number (Reduced, CX) counts data.
groups	List where names are groups labels and elements are patients labels corresponding to rownames in counts.
normalize	Boolean value; shall I normalize observed counts?
xlabels	Boolean value; shall I display x labels?

Value

A ggplot2 object.

Examples

```
data(plot_data_examples)
counts <- plot_data_examples[['groups.CX.plot']][['counts']]
groups <- plot_data_examples[['groups.CX.plot']][['groups']]
groupsCXPlot(counts=counts,groups=groups)
```

`groupsIDPlot` *groupsIDPlot*

Description

Plot observed Small Insertions and Deletions (ID) counts for different groups of patients.

Usage

```
groupsIDPlot(counts, groups, normalize = TRUE, xlabels = FALSE)
```

Arguments

`counts`	Matrix with Small Insertions and Deletions (ID) counts data.
`groups`	List where names are groups labels and elements are patients labels corresponding to rownames in counts.
`normalize`	Boolean value; shall I normalize observed counts?
`xlabels`	Boolean value; shall I display x labels?

Value

A ggplot2 object.

Examples

```
data(plot_data_examples)
counts <- plot_data_examples[['groups.ID.plot']][['counts']]
groups <- plot_data_examples[['groups.ID.plot']][['groups']]
groupsIDPlot(counts=counts,groups=groups)
```

groupsMNVPlot *groupsMNVPlot*

Description

Plot observed Multi-Nucleotide Variants (MNVs) counts for different groups of patients.

Usage

```
groupsMNVPlot(counts, groups, normalize = TRUE, xlabels = FALSE)
```

Arguments

counts	Matrix with Multi-Nucleotide Variants (MNVs) counts data.
groups	List where names are groups labels and elements are patients labels corresponding to rownames in counts.
normalize	Boolean value; shall I normalize observed counts?
xlabels	Boolean value; shall I display x labels?

Value

A ggplot2 object.

Examples

```
data(plot_data_examples)
counts <- plot_data_examples[['groups.MNV.plot']][['counts']]
groups <- plot_data_examples[['groups.MNV.plot']][['groups']]
groupsMNVPlot(counts=counts,groups=groups)
```

groupsSBSPlot *groupsSBSPlot*

Description

Plot observed Single Base Substitutions (SBS) counts for different groups of patients.

Usage

```
groupsSBSPlot(counts, groups, normalize = TRUE, xlabels = FALSE)
```

Arguments

counts	Matrix with Single Base Substitutions (SBS) counts data.
groups	List where names are groups labels and elements are patients labels corresponding to rownames in counts.
normalize	Boolean value; shall I normalize observed counts?
xlabels	Boolean value; shall I display x labels?

Value

A ggplot2 object.

Examples

```
data(plot_data_examples)
counts <- plot_data_examples[['groups.SBS.plot']][['counts']]
groups <- plot_data_examples[['groups.SBS.plot']][['groups']]
groupsSBSPlot(counts=counts,groups=groups)
```

---

id_example_reduced *A reduced version of the indel data for 3 samples in the format compatible with the import function*

Description

Reduced version of the dataset of counts of indels detected in 3 samples from Osorio, Fernando G., et al. (2018).

Usage

```
data(id_example_reduced)
```

Format

Reduced version of the counts of indels in the format compatible with the import function

Value

Reduced version of the counts of indels for 3 samples and 83 indel classes in the format compatible with the import function

Source

Cell Rep. 2018 Nov 27;25(9):2308-2316.e4 (10.1016/j.celrep.2018.11.014).

---

patients *Point mutations for 560 breast tumors*

Description

Dataset of counts of the point mutations detected in 560 breast tumors published in Nik-Zainal, Serena, et al. (2016).

Usage

```
data(patients)
```

Format

Counts of the point mutations

Value

Counts of point mutations for 560 tumors and 96 trinucleotides

Source

Nature. 2016 Jun 2;534(7605):47-54 (https://www.nature.com/articles/nature17676).

`patientsCNPlot` *patientsCNPlot*

Description

Plot Copy Number (CN) counts for a set of given patients.

Usage

```
patientsCNPlot(
  cn_data_counts,
  samples = rownames(cn_data_counts),
  freq = FALSE,
  xlabels = FALSE
)
```

Arguments

`cn_data_counts`	Copy Number counts matrix.
`samples`	Name of the samples. This should match a rownames in cn_data_counts
`freq`	Boolean value; shall I display rates instead of counts?
`xlabels`	Boolean value; shall I display x labels?

Value

A ggplot2 object.

Examples

```
data(plot_data_examples)
counts <- plot_data_examples[['patients.CN.plot']][['counts']]
patientsCNPlot(cn_data_counts=counts,samples=rownames(counts)[seq_len(2)])
```

patientsCXPlot *patientsCXPlot*

Description

Plot Copy Number (Reduced, CX) counts for a set of given patients.

Usage

```
patientsCXPlot(
  cn_data_counts,
  samples = rownames(cn_data_counts),
  freq = FALSE,
  xlabels = FALSE
)
```

Arguments

cn_data_counts	Copy Number counts matrix.
samples	Name of the samples. This should match a rownames in cn_data_counts
freq	Boolean value; shall I display rates instead of counts?
xlabels	Boolean value; shall I display x labels?

Value

A ggplot2 object.

Examples

```
data(plot_data_examples)
counts <- plot_data_examples[['patients.CX.plot']][['counts']]
patientsCXPlot(cn_data_counts=counts,samples=rownames(counts)[seq_len(2)])
```

patientsIDPlot *patientsIDPlot*

Description

Plot Small Insertions and Deletions (ID) counts for a set of given patients.

Usage

```
patientsIDPlot(
  id_data_counts,
  samples = rownames(id_data_counts),
  freq = FALSE,
  xlabels = FALSE
)
```

Arguments

id_data_counts	Small Insertions and Deletions counts matrix.
samples	Name of the samples. This should match a rownames in id_data_counts
freq	Boolean value; shall I display rates instead of counts?
xlabels	Boolean value; shall I display x labels?

Value

A ggplot2 object.

Examples

```
data(plot_data_examples)
counts <- plot_data_examples[['patients.ID.plot']][['counts']]
patientsIDPlot(id_data_counts=counts,samples=rownames(counts)[seq_len(2)])
```

---

patientsMNVPlot *patientsMNVPlot*

Description

Plot Multi-Nucleotide Variants (MNVs) counts for a set of given patients.

Usage

```
patientsMNVPlot(
  multi_nucleotides_counts,
  samples = rownames(multi_nucleotides_counts),
  freq = FALSE,
  xlabels = FALSE
)
```

Arguments

multi_nucleotides_counts	Multi-Nucleotide counts matrix.
samples	Name of the samples. This should match a rownames in multi_nucleotides_counts
freq	Boolean value; shall I display rates instead of counts?
xlabels	Boolean value; shall I display x labels?

Value

A ggplot2 object.

Examples

```
data(plot_data_examples)
counts <- plot_data_examples[['patients.MNV.plot']][['counts']]
patientsMNVPlot(multi_nucleotides_counts=counts,samples=rownames(counts)[seq_len(2)])
```

patientsSBSPlot *patientsSBSPlot*

Description

Plot Single Base Substitutions (SBS) counts for a set of given patients.

Usage

```
patientsSBSPlot(
  trinucleotides_counts,
  samples = rownames(trinucleotides_counts),
  freq = FALSE,
  xlabels = FALSE
)
```

Arguments

trinucleotides_counts	Trinucleotides counts matrix.
samples	Name of the samples. This should match a rownames in trinucleotides_counts.
freq	Boolean value; shall I display rates instead of counts?
xlabels	Boolean value; shall I display x labels?

Value

A ggplot2 object.

Examples

```
data(plot_data_examples)
counts <- plot_data_examples[['patients.SBS.plot']][['counts']]
patientsSBSPlot(trinucleotides_counts=counts,samples=rownames(counts)[seq_len(2)])
```

plot_data_examples *List data structure to run examples*

Description

List data structure to run examples.

Usage

```
data(plot_data_examples)
```

Format

List data structure to run examples

Value

List data structure to run examples

Source

List data structure to run examples.

---

sbs_assignments *SBS assignments for 560 breast tumors*

Description

SBS assignments from 560 breast tumors published in Nik-Zainal, Serena, et al. (2016).

Usage

```
data(sbs_assignments)
```

Format

SBS assignments inferred by RESOLVE from 560 breast tumors published in Nik-Zainal, Serena, et al. (2016)

Value

SBS assignments inferred by RESOLVE

Source

Nature. 2016 Jun 2;534(7605):47-54 (https://www.nature.com/articles/nature17676).

---

signaturesAssignment *signaturesAssignment*

Description

Perform the assignment of K somatic mutational signatures provided as input to samples given a set of observed counts x. This function can be used to estimate different types of mutational signatures such as: SBS (single base substitutions) and MNV (multi-nucleotide variant) (see Degasperi, Andrea, et al. 'Substitution mutational signatures in whole-genome–sequenced cancers in the UK population.' Science 376.6591 (2022): abl9283), CX (chromosomal instability) (see Drews, Ruben M., et al. 'A pan-cancer compendium of chromosomal instability.' Nature 606.7916 (2022): 976-983) and CN (copy number) signatures (see Steele, Christopher D., et al. 'Signatures of copy number alterations in human cancer.' Nature 606.7916 (2022): 984-991).

Usage

```
signaturesAssignment(x, beta)
```

Arguments

x	Counts matrix for a set of n patients and m categories. These can be, e.g., SBS, MNV, CN or CN counts; in the case of SBS it would be an n patients x 96 trinucleotides matrix.
beta	Matrix of the discovered signatures to be used for the assignment.

Value

A list with the discovered signatures. It includes 3 elements: alpha: matrix of the discovered exposure values. beta: matrix of the discovered signatures. unexplained_mutations: number of unexplained mutations per sample.

Examples

```
data(background)
data(patients)
set.seed(12345)
beta <- signaturesDecomposition(x = patients[seq_len(3),seq_len(2)],
                                K = 3,
                                background_signature = background[seq_len(2)],
                                nmf_runs = 2,
                                num_processes = 1)
set.seed(12345)
res <- signaturesAssignment(x = patients[seq_len(3),seq_len(2)], beta = beta$beta[[1]])
```

---

signaturesClustering *signaturesClustering*

Description

Perform signatures based clustering given the signatures assignments matrix alpha.

Usage

```
signaturesClustering(
  alpha,
  num_clusters = 1:10,
  num_processes = Inf,
  verbose = TRUE
)
```

Arguments

alpha	Signatures assignments matrix.
num_clusters	Range of number of clusters to be considered.
num_processes	Number of processes to be used during parallel execution. To execute in single process mode, this parameter needs to be set to either NA or NULL.
verbose	Boolean. Shall I print information messages?

Value

A list a clusters assignments for each number within the num_clusters range.

Examples

```
data(sbs_assignments)
set.seed(12345)
norm_alpha = (sbs_assignments$alpha / rowSums(sbs_assignments$alpha))
sbs_clustering = signaturesClustering(alpha = norm_alpha, num_clusters = 1:3,
                                      num_processes = 1, verbose = FALSE)
```

---

signaturesCNPlot *signaturesCNPlot*

Description

Plot the inferred Copy Number (CN) mutational signatures.

Usage

```
signaturesCNPlot(beta, useRowNames = FALSE, xlabels = FALSE)
```

Arguments

beta	Matrix with the inferred mutational signatures.
useRowNames	Boolean value; shall I use the rownames from beta as names for the signatures?
xlabels	Boolean value; shall I display x labels?

Value

A ggplot2 object.

Examples

```
data(plot_data_examples)
beta <- plot_data_examples[['signatures.CN.plot']][['beta']]
signaturesCNPlot(beta=beta)
```

signaturesCV *signaturesCV*

Description

Perform the assessment of different signaturesDecomposition solutions by cross-validation for K (beta, as estimated by signaturesDecomposition) somatic mutational signatures given a set of observations x and discovered signatures beta.

Usage

```
signaturesCV(
  x,
  beta,
  normalize_counts = FALSE,
  cross_validation_entries = 0.01,
  cross_validation_iterations = 5,
  cross_validation_repetitions = 100,
  num_processes = Inf,
  verbose = TRUE
)
```

Arguments

x	Counts matrix for a set of n patients and m categories. These can be, e.g., SBS, MNV, CN or CN counts; in the case of SBS it would be an n patients x 96 trinucleotides matrix.
beta	A set of inferred signatures as returned by signaturesDecomposition function.
normalize_counts	If true, the input counts matrix x is normalized such that the patients have the same number of mutation.
cross_validation_entries	Percentage of cells in the counts matrix to be replaced by 0s during cross-validation.
cross_validation_iterations	For each configuration, the first time the signatures are fitted form a matrix with a percentage of values replaced by 0s. This may result in poor fit/results. Then, we perform predictions of these entries and replace them with such predicted values. This parameter is the number of restarts to be performed to improve this estimate and obtain more stable solutions.
cross_validation_repetitions	Number of time cross-validation should be repeated. Higher values result in better estimate, but are computationally more expensive.
num_processes	Number of processes to be used during parallel execution. To execute in single process mode, this parameter needs to be set to either NA or NULL.
verbose	Boolean. Shall I print information messages?

Value

A list of 2 elements: estimates and summary. Here, cv_estimates reports the mean squared error for each configuration of performed cross-validation; rank_estimates reports mean and median values for each value of K.

Examples

```
data(background)
data(patients)
set.seed(12345)
sigs <- signaturesDecomposition(x = patients[seq_len(3),seq_len(2)],
                                K = 3:4,
                                background_signature = background[seq_len(2)],
                                nmf_runs = 2,
                                num_processes = 1)
set.seed(12345)
res <- signaturesCV(x = patients[seq_len(3),seq_len(2)],
                    beta = sigs$beta,
                    cross_validation_iterations = 2,
                    cross_validation_repetitions = 2,
                    num_processes = 1)
```

`signaturesCXPlot` *signaturesCXPlot*

Description

Plot the inferred Copy Number (Reduced, CX) mutational signatures.

Usage

```
signaturesCXPlot(beta, useRowNames = FALSE, xlabels = FALSE)
```

Arguments

`beta`	Matrix with the inferred mutational signatures.
`useRowNames`	Boolean value; shall I use the rownames from beta as names for the signatures?
`xlabels`	Boolean value; shall I display x labels?

Value

A ggplot2 object.

Examples

```
data(plot_data_examples)
beta <- plot_data_examples[['signatures.CX.plot']][['beta']]
signaturesCXPlot(beta=beta)
```

signaturesDecomposition *signaturesDecomposition*

Description

Perform signatures discovery and rank estimation for a range of K somatic mutational signatures given a set of observed counts x. This function can be used to estimate different types of mutational signatures such as: SBS (single base substitutions) and MNV (multi-nucleotide variant) (see Degasperi, Andrea, et al. 'Substitution mutational signatures in whole-genome–sequenced cancers in the UK population.' Science 376.6591 (2022): abl9283), CX (chromosomal instability) (see Drews, Ruben M., et al. 'A pan-cancer compendium of chromosomal instability.' Nature 606.7916 (2022): 976-983) and CN (copy number) signatures (see Steele, Christopher D., et al. 'Signatures of copy number alterations in human cancer.' Nature 606.7916 (2022): 984-991).

Usage

```
signaturesDecomposition(
  x,
  K,
  background_signature = NULL,
  normalize_counts = FALSE,
  nmf_runs = 100,
  num_processes = Inf,
  verbose = TRUE
)
```

Arguments

x	Counts matrix for a set of n patients and m categories. These can be, e.g., SBS, MNV, CN or CN counts; in the case of SBS it would be an n patients x 96 trinucleotides matrix.
K	Either one value or a range of numeric values (each of them greater than 0) indicating the number of signatures to be considered.
background_signature	Background signature to be used.
normalize_counts	If true, the input counts matrix x is normalized such that the patients have the same number of mutation.
nmf_runs	Number of iteration (minimum 1) of NMF to be performed for a robust estimation of beta.
num_processes	Number of processes to be used during parallel execution. To execute in single process mode, this parameter needs to be set to either NA or NULL.
verbose	Boolean. Shall I print information messages?

Value

A list with the discovered signatures and related rank measures. It includes 5 elements: alpha: list of matrices of the discovered exposure values for each possible rank in the range K. beta: list of

matrices of the discovered signatures for each possible rank in the range K. unexplained_mutations: number of unexplained mutations per sample. cosine_similarity: cosine similarity comparing input data x and predictions for each rank in the range K. measures: a data.frame containing the quality measures for each possible rank in the range K.

Examples

```
data(background)
data(patients)
set.seed(12345)
res <- signaturesDecomposition(x = patients[seq_len(3),seq_len(2)],
                               K = 3:4,
                               background_signature = background[seq_len(2)],
                               nmf_runs = 2,
                               num_processes = 1)
```

`signaturesIDPlot` *signaturesIDPlot*

Description

Plot the inferred Small Insertions and Deletions (ID) mutational signatures.

Usage

```
signaturesIDPlot(beta, useRowNames = FALSE, xlabels = FALSE)
```

Arguments

`beta`	Matrix with the inferred mutational signatures.
`useRowNames`	Boolean value; shall I use the rownames from beta as names for the signatures?
`xlabels`	Boolean value; shall I display x labels?

Value

A ggplot2 object.

Examples

```
data(plot_data_examples)
beta <- plot_data_examples[['signatures.ID.plot']][['beta']]
signaturesIDPlot(beta=beta)
```

signaturesMNVPlot *signaturesMNVPlot*

Description

Plot the inferred Multi-Nucleotide Variants (MNVs) mutational signatures.

Usage

```
signaturesMNVPlot(beta, useRowNames = FALSE, xlabels = FALSE)
```

Arguments

beta	Matrix with the inferred mutational signatures.
useRowNames	Boolean value; shall I use the rownames from beta as names for the signatures?
xlabels	Boolean value; shall I display x labels?

Value

A ggplot2 object.

Examples

```
data(plot_data_examples)
beta <- plot_data_examples[['signatures.MNV.plot']][['beta']]
signaturesMNVPlot(beta=beta)
```

signaturesSBSPlot *signaturesSBSPlot*

Description

Plot the inferred Single Base Substitutions (SBS) mutational signatures.

Usage

```
signaturesSBSPlot(beta, useRowNames = FALSE, xlabels = FALSE)
```

Arguments

beta	Matrix with the inferred mutational signatures.
useRowNames	Boolean value; shall I use the rownames from beta as names for the signatures?
xlabels	Boolean value; shall I display x labels?

Value

A ggplot2 object.

Examples

```
data(plot_data_examples)
beta <- plot_data_examples[['signatures.SBS.plot']][['beta']]
signaturesSBSPlot(beta=beta)
```

---

signaturesSignificance *signaturesSignificance*

Description

Perform a robust estimation of alpha coefficients by bootstrap to reach a certain level of cosine similarity given a set of observed counts x and discovered signatures beta.

Usage

```
signaturesSignificance(
  x,
  beta,
  cosine_thr = 0.95,
  min_contribution = 0.05,
  pvalue_thr = 0.05,
  nboot = 100,
  num_processes = Inf,
  verbose = TRUE
)
```

Arguments

x	Counts matrix for a set of n patients and m categories. These can be, e.g., SBS, MNV, CN or CN counts; in the case of SBS it would be an n patients x 96 trinucleotides matrix.
beta	Discovered signatures to be used for the fit of alpha.
cosine_thr	Level of cosine similarity to be reached for the fit of alpha.
min_contribution	Minimum contribution of a signature to be considered significant.
pvalue_thr	Pvalue level to be used to assess significance.
nboot	Number of bootstrap iterations to be performed.
num_processes	Number of processes to be used during parallel execution. To execute in single process mode, this parameter needs to be set to either NA or NULL.
verbose	Boolean. Shall I print information messages?

Value

A list with the bootstrap estimates. It includes 5 elements: alpha: matrix of the discovered exposure values considering significant signatures as estimated by bootstrap. beta: matrix of the discovered signatures. unexplained_mutations: number of unexplained mutations per sample. goodness_fit: vector reporting cosine similarities between predictions and observations. bootstrap_estimates: list of matrices reporting results by bootstrap estimates.

Examples

```
data(background)
data(patients)
set.seed(12345)
beta <- signaturesDecomposition(x = patients[seq_len(3),seq_len(2)],
                                K = 3:4,
                                background_signature = background[seq_len(2)],
                                nmf_runs = 2,
                                num_processes = 1)
set.seed(12345)
res <- signaturesSignificance(x = patients[seq_len(3),seq_len(2)],
                              beta = beta$beta[[1]],
                              cosine_thr = 0.95,
                              min_contribution = 0.05,
                              pvalue_thr = 0.05,
                              nboot = 5,
                              num_processes = 1)
```

---

ssm560_reduced *A reduced version of the point mutations for 560 breast tumors in the format compatible with the import function*

Description

Reduced version of the dataset of counts of the point mutations detected in 560 breast tumors published in Nik-Zainal, Serena, et al. (2016).

Usage

```
data(ssm560_reduced)
```

Format

Reduced version of the counts of the point mutations in the format compatible with the import function

Value

Reduced version of the counts of point mutations for 560 tumors and 96 trinucleotides in the format compatible with the import function

Source

Nature. 2016 Jun 2;534(7605):47-54 (https://www.nature.com/articles/nature17676).

Index