Package ‘corral’

May 25, 2026

Title Correspondence Analysis for Single Cell Data

Version 1.23.0

Date 2023-02-09

Description Correspondence analysis (CA) is a matrix factorization method, and is similar to principal components analysis (PCA). Whereas PCA is designed for application to continuous, approximately normally distributed data, CA is appropriate for non-negative, count-based data that are in the same additive scale. The corral package implements CA for dimensionality reduction of a single matrix of single-cell data, as well as a multi-table adaptation of CA that leverages data-optimized scaling to align data generated from different sequencing platforms by projecting into a shared latent space. corral utilizes sparse matrices and a fast implementation of SVD, and can be called directly on Bioconductor objects (e.g., SingleCellExperiment) for easy pipeline integration. The package also includes additional options, including variations of CA to address overdispersion in count data (e.g., Freeman-Tukey chi-squared residual), as well as the option to apply CA-style processing to continuous data (e.g., proteomic TOF intensities) with the Hellinger distance adaptation of CA.

Imports ggplot2, ggthemes, grDevices, gridExtra, irlba, Matrix, methods, MultiAssayExperiment, pals, reshape2, SingleCellExperiment, SummarizedExperiment, transport

Suggests ade4, BiocStyle, CellBench, DuoClustering2018, knitr, rmarkdown, scater, testthat

License GPL-2

RoxygenNote 7.1.2

VignetteBuilder knitr

biocViews BatchEffect, DimensionReduction, GeneExpression, Preprocessing, PrincipalComponent, Sequencing, SingleCell, Software, Visualization

Encoding UTF-8

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

git_branch devel

git_last_commit 757a1c2

git_last_commit_date 2026-04-28

Repository Bioconductor 3.24

Date/Publication 2026-05-25

Author Lauren Hsu [aut, cre] (ORCID: <https://orcid.org/0000-0002-6035-7381>),
Aedin Culhane [aut] (ORCID: <https://orcid.org/0000-0002-1395-9734>)

Maintainer Lauren Hsu <lrnshoe@gmail.com>

Contents

add_embeddings2scelist 2
all_are 3
biplot_corral 4
compsvd 5
corralm_matlist 6
corral_mat 8
corral_preproc 10
earthmover_dist 12
get_pct_var_exp_svd 12
get_weights 13
list2mat 14
na2zero 14
obs2probs 15
pairwise_rv 15
plot_embedding 16
plot_embedding_sce 17
rv 19
scal_var 20
scal_var_mat 21
sce2matlist 21
trim_matdist 22
var_stabilize 23

Index **24**

`add_embeddings2scelist` *Add embeddings to list of SCEs*

Description

Add embeddings to list of SCEs

Usage

```
add_embeddings2scelist(scelist, embeddings, slotname = "corralm")
```

Arguments

`scelist`	list of SingleCellExperiments; to which the corresponding embeddings should be added
`embeddings`	matrix; the embeddings outputted from a dimension reduction, e.g. `corralm`. Rows in this table correspond to columns in the SCEs in `scelist` (if all the SCEs were column-bound), and row indices should correspond to cells.
`slotname`	character; name of the slot for the reduced dim embedding; defaults to `corral`

Value

list of SingleCellExperiments with respective embeddings stored in them

Examples

```
library(DuoClustering2018)
sce <- sce_full_Zhengmix4eq()
scelist <- list(sce,sce)
embeddings <- matrix(sample(seq(0,20,1),dim(sce)[2]*6,replace = TRUE),nrow = dim(sce)[2]*2)
scelist <- add_embeddings2scelist(scelist, embeddings)
```

`all_are` *all_are*

Description

Checks if all elements of a list or List are of a (single) particular type `typechar`

Usage

```
all_are(inplist, typechar)
```

Arguments

`inplist`	list or List to be checked
`typechar`	char of the type to check for

Value

boolean, for whether the elements of `inplist` are all `typechar`

Examples

```
x <- list(1,2)
all_are(x,'numeric')
all_are(x,'char')

y <- list(1,2,'c')
all_are(y,'numeric')
all_are(y,'char')
```

`biplot_corral` *Generate biplot for corral object*

Description

Generate biplot for corral object

Usage

```
biplot_corral(
  corral_obj,
  color_vec,
  text_vec,
  feat_name = "(genes)",
  nfeat = 20,
  xpc = 1,
  plot_title = "Biplot",
  text_size = 2,
  xjitter = 0.005,
  yjitter = 0.005,
  coords = c("svd", "PC", "SC")
)
```

Arguments

`corral_obj`	list outputted by the `corral` function
`color_vec`	vector; length should correspond to the number of rows in v of `corral_obj`, and each element of the vector classifies that cell (entry) in the embedding to that particular class, which will be colored the same. (e.g., cell type)
`text_vec`	vector; length should correspond to the number of rows in u of `corral_obj`, and each element of the vector is the label for the respective feature that would show on the biplot.
`feat_name`	char; the label will in the legend. Defaults to `(genes)`.
`nfeat`	int; the number of features to include. The function will first order them by distance from origin in the selected dimensions, then select the top n to be displayed.
`xpc`	int; which PC to put on the x-axis (defaults to 1)
`plot_title`	char; title of plot (defaults to *Biplot*)
`text_size`	numeric; size of the feature labels given in `text_vec` (defaults to 2; for `ggplot2`)
`xjitter`	numeric; the amount of jitter for the text labels in x direction (defaults to .005; for `ggplot2`)
`yjitter`	numeric; the amount of jitter for the text labels in y direction (defaults to .005; for `ggplot2`)
`coords`	char; indicator for sets of coordinates to use. `svd` plots the left and right singular vectors as outputted by SVD (`u` and `v`), which `PC` and `SC` use the principal and standard coordinates, respectively (defaults to `svd`)

Value

ggplot2 object of the biplot

Examples

```
library(DuoClustering2018)
library(SingleCellExperiment)
zm4eq.sce <- sce_full_Zhengmix4eq()
zm4eq.countmat <- counts(zm4eq.sce)
zm4eq.corral_obj <- corral(zm4eq.countmat)
gene_names <- rowData(zm4eq.sce)$symbol
ctvec <- zm4eq.sce$phenoid

biplot_corral(corral_obj = zm4eq.corral_obj, color_vec = ctvec, text_vec = gene_names)
```

`compsvd` *compsvd: Compute Singular Value Decomposition (SVD)*

Description

Computes SVD.

Usage

```
compsvd(mat, method = c("irl", "svd"), ncomp = 30, ...)
```

Arguments

`mat`	matrix, pre-processed input; can be sparse or full (pre-processing can be performed using `corral_preproc` from this package)
`method`	character, the algorithm to be used for svd. Default is irl. Currently supports 'irl' for irlba::irlba or 'svd' for stats::svd
`ncomp`	numeric, number of components; Default is 30
`...`	(additional arguments for methods)

Value

SVD result - a list with the following elements:

`d` a vector of the diagonal singular values of the input `mat`. Note that using `svd` will result in the full set of singular values, while `irlba` will only compute the first `ncomp` singular values.

`u` a matrix of with the left singular vectors of `mat` in the columns

`v` a matrix of with the right singular vectors of `mat` in the columns

`eigsum` sum of the eigenvalues, for calculating percent variance explained

Examples

```
mat <- matrix(sample(0:10, 2500, replace=TRUE), ncol=50)
compsvd(mat, method = 'irl', ncomp = 5)
```

`corralm_matlist` *Multi-table correspondence analysis (list of matrices)*

Description

This multi-table adaptation of correpondence analysis applies the same scaling technique and enables data alignment by finding a set of embeddings for each dataset within shared latent space.

Usage

```
corralm_matlist(
  matlist,
  method = c("irl", "svd"),
  ncomp = 30,
  rtype = c("indexed", "standardized", "hellinger", "freemantukey", "pearson"),
  vst_mth = c("none", "sqrt", "freemantukey", "anscombe"),
  rw_contrib = NULL,
  ...
)

corralm_sce(
  sce,
  splitby,
  method = c("irl", "svd"),
  ncomp = 30,
  whichmat = "counts",
  fullout = FALSE,
  rw_contrib = NULL,
  ...
)

corralm(inp, whichmat = "counts", fullout = FALSE, ...)

## S3 method for class 'corralm'
print(x, ...)
```

Arguments

`matlist`	(for `corralm_matlist`) list of input matrices; input matrices should be counts (raw or log). Matrices should be aligned row-wise by common features (either by sample or by gene)
`method`	character, the algorithm to be used for svd. Default is irl. Currently supports 'irl' for irlba::irlba or 'svd' for stats::svd
`ncomp`	numeric, number of components; Default is 30
`rtype`	character indicating what type of residual should be computed; options are ‘"indexed"‘, ‘"standardized"‘ (or ‘"pearson"‘ is equivalent), ‘"freemantukey"‘, and ‘"hellinger"‘; defaults to ‘"standardized"‘ for `corral` and ‘"indexed"‘ for `corralm`. ‘"indexed"‘, ‘"standardized"‘, and ‘"freemantukey"‘ compute the respective chi-squared residuals and are appropriate for count data. The ‘"hellinger"‘ option is appropriate for continuous data.

`vst_mth`	character indicating whether a variance-stabilizing transform should be applied prior to calculating chi-squared residuals; defaults to ‘"none"‘
`rw_contrib`	numeric vector, same length as the matlist. Indicates the weight that each dataset should contribute to the row weights. When set to NULL the row weights are *not* combined and each matrix is scaled independently (i.e., using their observed row weights, respectively). When set to a vector of all the same values, this is equivalent to taking the mean. Another option is to the number of observations per matrix to create a weighted mean. Regardless of input scale, row weights for each table must sum to 1 and thus are scaled. When this option is specified (i.e., not ‘NULL‘), the ‘rtype‘ argument will automatically be set to ‘standardized‘, and whatever argument is given will be ignored.
`...`	(additional arguments for methods)
`sce`	(for `corralm_sce`) SingleCellExperiment; containing the data to be integrated. Default is to use the counts, and to include all of the data in the integration. These can be changed by passing additional arguments. See `sce2matlist` function documentation for list of available parameters.
`splitby`	character; name of the attribute from `colData` that should be used to separate the SCE.
`whichmat`	char, when using SingleCellExperiment or other SummarizedExperiment, can be specified. default is 'counts'.
`fullout`	boolean; whether the function will return the full `corralm` output as a list, or a SingleCellExperiment; defaults to SingleCellExperiment (`FALSE`). To get back the `corralm_matlist`-style output, set this to `TRUE`.
`inp`	list of matrices (any type), a `SingleCellExperiment`, list of `SingleCellExperiments`, list of `SummarizedExperiments`, or `MultiAssayExperiment`. If using `SingleCellExperiment` or `SummarizedExperiment`, then include the `whichmat` argument to specify which slot to use (defaults to `counts`). Additionally, if it is one `SingleCellExperiment`, then it is also necessary to include the `splitby` argument to specify the batches. For a `MultiAssayExperiment`, it will take the intersect of the features across all the assays, and use those to match the matrices; to use a different subset, select desired subsets then call `corral`
`x`	(print method) corralm object; the list output from `corralm_matlist`

Details

`corralm` is a wrapper for `corralm_matlist` and `corralm_sce`, and can be called on any of the acceptable input types (see `inp` below).

Value

When run on a list of matrices, a list with the correspondence analysis matrix decomposition result, with indices corresponding to the concatenated matrices (in order of the list):

`d`	a vector of the diagonal singular values of the input `mat` (from SVD output)
`u`	a matrix of with the left singular vectors of `mat` in the columns (from SVD output)
`v`	a matrix of with the right singular vectors of `mat` in the columns. When cells are in the columns, these are the cell embeddings. (from SVD output)
`eigsum`	sum of the eigenvalues for calculating percent variance explained

For SingleCellExperiment input, returns the SCE with embeddings in the reducedDim slot 'corralm'

For a list of `SingleCellExperiment`s, returns a list of the SCEs with the embeddings in the respective `reducedDim` slot 'corralm'

.

Examples

```
listofmats <- list(matrix(sample(seq(0,20,1),1000,replace = TRUE),nrow = 25),
                   matrix(sample(seq(0,20,1),1000,replace = TRUE),nrow = 25))
result <- corralm_matlist(listofmats)
library(DuoClustering2018)
library(SingleCellExperiment)
sce <- sce_full_Zhengmix4eq()[1:100,sample(1:3500,100,replace = FALSE)]
colData(sce)$Method <- matrix(sample(c('Method1','Method2'),100,replace = TRUE))
result <- corralm_sce(sce, splitby = 'Method')


listofmats <- list(matrix(sample(seq(0,20,1),1000,replace = TRUE),nrow = 20),
                   matrix(sample(seq(0,20,1),1000,replace = TRUE),nrow = 20))
corralm(listofmats)

library(DuoClustering2018)
library(SingleCellExperiment)
sce <- sce_full_Zhengmix4eq()[seq(1,100,1),sample(seq(1,3500,1),100,replace = FALSE)]
colData(sce)$Method <- matrix(sample(c('Method1','Method2'),100,replace = TRUE))
result <- corralm(sce, splitby = 'Method')

# default print method for corralm objects
```

`corral_mat` *corral: Correspondence analysis on a single matrix*

Description

corral can be used for dimension reduction to find a set of low-dimensional embeddings for a count matrix.

`corral` is a wrapper for `corral_mat` and `corral_sce`, and can be called on any of the acceptable input types.

Usage

```
corral_mat(
  inp,
  method = c("irl", "svd"),
  ncomp = 30,
  row.w = NULL,
  col.w = NULL,
  rtype = c("standardized", "indexed", "hellinger", "freemantukey", "pearson"),
  vst_mth = c("none", "sqrt", "freemantukey", "anscombe"),
  ...
)
```

```
corral_sce(
  inp,
  method = c("irl", "svd"),
  ncomp = 30,
  whichmat = "counts",
  fullout = FALSE,
  subset_row = NULL,
  ...
)

corral(inp, ...)

## S3 method for class 'corral'
print(x, ...)
```

Arguments

`inp`	matrix (any type), `SingleCellExperiment`, or `SummarizedExperiment`. If using `SingleCellExperiment` or `SummarizedExperiment`, then include the `whichmat` argument to specify which slot to use (defaults to `counts`).
`method`	character, the algorithm to be used for svd. Default is irl. Currently supports 'irl' for irlba::irlba or 'svd' for stats::svd
`ncomp`	numeric, number of components; Default is 30
`row.w`	numeric vector; the row weights to use in chi-squared scaling. Defaults to ‘NULL‘, in which case row weights are computed from the input matrix.
`col.w`	numeric vector; the column weights to use in chi-squared scaling. For instance, size factors could be given here. Defaults to ‘NULL‘, in which case column weights are computed from the input matrix.
`rtype`	character indicating what type of residual should be computed; options are ‘"indexed"‘, ‘"standardized"‘ (or ‘"pearson"‘ is equivalent), ‘"freemantukey"‘, and ‘"hellinger"‘; defaults to ‘"standardized"‘ for `corral` and ‘"indexed"‘ for `corralm`. ‘"indexed"‘, ‘"standardized"‘, and ‘"freemantukey"‘ compute the respective chi-squared residuals and are appropriate for count data. The ‘"hellinger"‘ option is appropriate for continuous data.
`vst_mth`	character indicating whether a variance-stabilizing transform should be applied prior to calculating chi-squared residuals; defaults to ‘"none"‘
`...`	(additional arguments for methods)
`whichmat`	character; defaults to `counts`, can also use `logcounts` or `normcounts` if stored in the sce object
`fullout`	boolean; whether the function will return the full `corral` output as a list, or a SingleCellExperiment; defaults to SingleCellExperiment (`FALSE`). To get back the `corral_mat`-style output, set this to `TRUE`.
`subset_row`	numeric, character, or boolean vector; the rows to include in corral, as indices (numeric), rownames (character), or with booleans (same length as the number of rows in the matrix). If this parameter is `NULL`, then all rows will be used.
`x`	(print method) corral object; the list output from `corral_mat`

Value

When run on a matrix, a list with the correspondence analysis matrix decomposition result:

- `d` a vector of the diagonal singular values of the input `mat` (from SVD output)
- `u` a matrix of with the left singular vectors of `mat` in the columns (from SVD output)
- `v` a matrix of with the right singular vectors of `mat` in the columns. When cells are in the columns, these are the cell embeddings. (from SVD output)
- `eigsum` sum of the eigenvalues for calculating percent variance explained
- `SCu and SCv` standard coordinates, left and right, respectively
- `PCu and PCv` principal coordinates, left and right, respectively

When run on a `SingleCellExperiment`, returns a SCE with the embeddings (PCv from the full corral output) in the `reducedDim` slot `corral` (default). Also can return the same output as `corral_mat` when `fullout` is set to `TRUE`.

For matrix and `SummarizedExperiment` input, returns list with the correspondence analysis matrix decomposition result (u,v,d are the raw svd output; SCu and SCv are the standard coordinates; PCu and PCv are the principal coordinates)

For `SummarizedExperiment` input, returns the same as for a matrix.

.

Examples

```
mat <- matrix(sample(0:10, 5000, replace=TRUE), ncol=50)
result <- corral_mat(mat)
result <- corral_mat(mat, method = 'irl', ncomp = 5)

library(DuoClustering2018)
sce <- sce_full_Zhengmix4eq()[1:100,1:100]
result_1 <- corral_sce(sce)
result_2 <- corral_sce(sce, method = 'svd')
result_3 <- corral_sce(sce, method = 'irl', ncomp = 30, whichmat = 'logcounts')


library(DuoClustering2018)
sce <- sce_full_Zhengmix4eq()[1:100,1:100]
corral_sce <- corral(sce,whichmat = 'counts')

mat <- matrix(sample(0:10, 500, replace=TRUE), ncol=25)
corral_mat <- corral(mat, ncomp=5)

mat <- matrix(sample(1:100, 10000, replace = TRUE), ncol = 100)
corral(mat)
```

`corral_preproc` *Preprocess a matrix for SVD to perform Correspondence Analysis (CA)*

Description

This function performs the row and column scaling pre-processing operations, prior to SVD, for the corral methods. See `corral` for single matrix correspondence analysis and `corralm` for multi-matrix correspondence analysis.

Usage

```
corral_preproc(
  inp,
  rtype = c("standardized", "indexed", "hellinger", "freemantukey", "pearson"),
  vst_mth = c("none", "sqrt", "freemantukey", "anscombe"),
  powdef_alpha = NULL,
  row.w = NULL,
  col.w = NULL,
  smooth = FALSE,
  ...
)
```

Arguments

`inp`	matrix, numeric, counts or logcounts; can be sparse Matrix or matrix
`rtype`	character indicating what type of residual should be computed; options are ‘"indexed"‘, ‘"standardized"‘ (or ‘"pearson"‘ is equivalent), ‘"freemantukey"‘, and ‘"hellinger"‘; defaults to ‘"standardized"‘ for `corral` and ‘"indexed"‘ for `corralm`. ‘"indexed"‘, ‘"standardized"‘, and ‘"freemantukey"‘ compute the respective chi-squared residuals and are appropriate for count data. The ‘"hellinger"‘ option is appropriate for continuous data.
`vst_mth`	character indicating whether a variance-stabilizing transform should be applied prior to calculating chi-squared residuals; defaults to ‘"none"‘
`powdef_alpha`	numeric for the power that should be applied if using power deflation. Must be in (0,1), and if provided a number outside this range, will be ignored. Defaults to ‘NULL‘ which does not perform this step.
`row.w`	numeric vector; Default is `NULL`, to compute row.w based on `inp`. Use this parameter to replace computed row weights with custom row weights
`col.w`	numeric vector; Default is `NULL`, to compute col.w based on `inp`. Use this parameter to replace computed column weights with custom column weights
`smooth`	logical; Whether or not to perform the additional smoothing step with ‘trim_matdist‘. Default is `FALSE`. Incompatible with ‘powdef_alpha‘, so that parameter takes precedence over this one.
`...`	(additional arguments for methods)

Value

matrix, processed for input to `compsvd` to finish CA routine

Examples

```
mat <- matrix(sample(0:10, 500, replace=TRUE), ncol=25)
mat_corral <- corral_preproc(mat)
corral_output <- compsvd(mat_corral, ncomp = 5)
```

`earthmover_dist` *Earthmover distance (and general Wasserstein distance)*

Description

i.e., wasserstein distance with L1 (p_param = 1); can also use other penalties > 1 (Not technically earthmover distance if using other p_param values)

Usage

```
earthmover_dist(batch1, batch2, whichdim = 1, numbins = 100, p_param = 1)
```

Arguments

`batch1`	matrix; subset of observations from an embedding correponding to some attribute (e.g., batch or phenotype)
`batch2`	matrix; subset of observations from an embedding correponding to some attribute (e.g., batch or phenotype)
`whichdim`	int; which dimension (i.e., column) from the embeddings is used. defaults on first
`numbins`	int; number of bins for the probability discretization (defaults to 100)
`p_param`	int; penalty parameter for general Wasserstein distance. Defaults to 1, which corresonds to earthmover.

Value

num; the distance

Examples

```
# To compare distributions of reduced dimension values to assess similarity,
# e.g. as a metric for batch integration
embedding <- matrix(sample(x = seq(0,10,.1),1000, replace = TRUE),ncol = 5)
batch <- matrix(sample(c(1,2),200, replace = TRUE))
earthmover_dist(embedding[which(batch == 1),],embedding[which(batch == 2),])
```

`get_pct_var_exp_svd` *Compute percent of variance explained*

Description

Compute percent of variance explained

Usage

```
get_pct_var_exp_svd(thissvd, preproc_mat = thissvd$d)
```

Arguments

`thissvd`	list outputted from an svd function (svd, irlba; can also take output from `corral_mat` and `corralm_matlist`)
`preproc_mat`	matrix of pre-processed values (optional) - important to include if the svd is only partial as this is used to compute the sum of eigenvalues

Value

vector of percent variance explained values, indexed by PC

Examples

```
mat <- matrix(sample(seq(0,20,1),100,replace = TRUE),nrow = 10)
my_svd <- svd(mat)
get_pct_var_exp_svd(my_svd) # this works if my_svd is a full svd
my_irl <- irlba::irlba(mat,nv = 2)
get_pct_var_exp_svd(my_irl, preproc_mat = mat) # ... otherwise use this
```

`get_weights` *Get weights*

Description

Computes row weights and column weights

Usage

```
get_weights(inp_mat)
```

Arguments

`inp_mat`	matrix for which weights should be calculated (sparse or full)

Value

list of 2 elements: 'row.w' and 'col.w' contain the row and column weights respectively

Examples

```
mat <- matrix(sample(seq(0,20,1),100,replace = TRUE),nrow = 10)
ws <- get_weights(mat)
```

`list2mat` *List to Matrix*

Description

List to Matrix

Usage

```
list2mat(matlist, direction = c("c", "r")[1])
```

Arguments

`matlist`	list of matrices to concatenate
`direction`	character, r or c, to indicate whether should be row-wise (i.e., rbind to match on columns) or column-wise (i.e., cbind to match on rows). Defaults to columnwise (matching on rows) to match convention of SingleCellExperiments

Value

matrix

Examples

```
listofmats <- list(matrix(sample(seq(0,20,1),100,replace = TRUE),nrow = 10),
                   matrix(sample(seq(0,20,1),1000,replace = TRUE),nrow = 10))
newmat <- list2mat(listofmats) # to "cbind" them
listofmats_t <- lapply(listofmats,t)
newmat_t <- list2mat(listofmats_t, 'r') # to "rbind" them
```

`na2zero` *Set na to 0*

Description

Set na to 0

Usage

```
na2zero(x)
```

Arguments

`x`	matrix of values for which na values should be changed to 0

Value

matrix, where na values are set to 0

Examples

```
x <- matrix(sample(0:10, 5000, replace = TRUE), ncol = 25)
x[sample(1:5000, 10)] <- NA

na2zero(x)
```

`obs2probs` *Observations –> discrete probabilities*

Description

usage: embedding <- matrix(sample(x = seq(0,10,.1),200, replace = TRUE)) disc_probs <- obs2probs(embedding)

Usage

```
obs2probs(obs, numbins = 100, startbin = min(obs), endbin = max(obs) + 1e-05)
```

Arguments

`obs`	vector of numeric, with the observations
`numbins`	int, the number of evenly sized bins to discretize the observations to
`startbin`	numeric, the starting value for the smallest bin. Defaults to taking the minimum of obs
`endbin`	numeric, the ending value for the largest bin. Defaults to taking the maximum of obs (plus a tiny decimal to ensure full range of obs is captured)

Value

dataframe, results has rows corresponding to each bin with columns for probability ('prob'), cumulative frequency ('cumfreq'), and frequency ('freq') of observations falling into that bin. The 'bins' column indicates the end of the bin (start is the preceding column)

`pairwise_rv` *Pairwise rv coefficient*

Description

Pairwise rv coefficient

Usage

```
pairwise_rv(matlist)
```

Arguments

`matlist`	list of matrices (or matrix-like; see `rv` function) for which to compute pairwise RV coefficients

Value

matrix of the pairwise coefficients

Examples

```
a <- matrix(sample(1:10,100,TRUE), nrow = 10)
b <- matrix(sample(1:10,50,TRUE), nrow = 5)
c <- matrix(sample(1:10,20,TRUE), nrow = 2)

matlist <- list(a,b,c)
pairwise_rv(matlist)
pairwise_rv(lapply(matlist, t))
```

`plot_embedding` *Plot selected PCs from an embedding*

Description

Plot selected PCs from an embedding

Usage

```
plot_embedding(
  embedding,
  xpc = 1,
  ypc = xpc + 1,
  plot_title = paste0("Dim", xpc, " by Dim", ypc),
  color_vec = NULL,
  color_title = NULL,
  ellipse_vec = NULL,
  facet_vec = NULL,
  ptsize = 0.8,
  saveplot = FALSE,
  plotfn = paste(plot_title, xpc, sep = "_"),
  showplot = TRUE,
  returngg = FALSE,
  color_pal_vec = NULL,
  dimname = "Dim"
)
```

Arguments

`embedding`	matrix or other tabular format where columns correspond to PCs and rows correspond to cells (entries). `corral` and `corralm` objects are also accepted.
`xpc`	int; which PC to put on the x-axis (defaults to 1)
`ypc`	int; which PC to put on the y-axis (defaults to the one after `xpc`)
`plot_title`	char; title of plot (defaults to titling based on `xpc` and `ypc`)
`color_vec`	vector; length should correspond to the number of rows in embedding, and each element of the vector classifies that cell (entry) in the embedding to that particular class, which will be colored the same. (e.g., this could be indicating which batch each cell is from)

`color_title`	char; what attribute the colors represent
`ellipse_vec`	vector; length should correspond to the number of rows in embedding, and each element of the vector classifies that cell (entry) in the embedding to that particular class, and elements of the same class will be circled in an ellipse. (e.g., this could be indicating the cell type or cell line; works best for attributes intended to be compact)
`facet_vec`	vector; length should correspond to the number of rows in embedding, and each element of the vector classifies that cell (entry) in the embedding to that particular class. Plot will be faceted by this attribute.
`ptsize`	numeric; the size of the points as passed to `geom_point()`. Defaults to 0.8.
`saveplot`	boolean; whether or not to save the plot, defaults `FALSE`
`plotfn`	char; what the filename is to be called. (defaults to making a name based on `plot_title` and `xpc`)
`showplot`	boolean; whether or not to show the plot, defaults `TRUE`
`returngg`	boolean; whether or not to return a `ggplot2` object, defaults `FALSE`
`color_pal_vec`	char; hex codes for the color palette to be used. Default is to use the ggthemes few for plots with less than 9 colors, and to use/"stretch" pals polychrome if more colors are needed.
`dimname`	char; the name of the dimensions. defaults to "Dim"

Value

default none; options to display plot (`showplot`), save plot (`saveplot`), and/or return `ggplot2` object (`returngg`)

Examples

```
listofmats <- list(matrix(sample(seq(0,20,1),1000,replace = TRUE),nrow = 20),
                   matrix(sample(seq(0,20,1),1000,replace = TRUE),nrow = 20))
corralm_obj <- corralm(listofmats, ncomp = 5)
embed_mat <- corralm_obj$v
cell_type_vec <- sample(c('type1','type2','type3'),100,replace = TRUE)
plot_embedding(embedding = embed_mat,
               xpc = 1,
               plot_title = 'corralm plot',
               color_vec = cell_type_vec,
               color_title = 'cell type',
               saveplot = FALSE)

# or, call directly on the corralm object
plot_embedding(corralm_obj)
```

`plot_embedding_sce` *Plot selected PCs from an embedding saved in a SingleCellExperiment object*

Description

Plot selected PCs from an embedding saved in a SingleCellExperiment object

Usage

```
plot_embedding_sce(
  sce,
  which_embedding,
  color_attr = NULL,
  color_title = color_attr,
  ellipse_attr = NULL,
  facet_attr = NULL,
  ...
)
```

Arguments

`sce`	`SingleCellExperiment` object; contains the embedding within the `reducedDim` slot
`which_embedding`	character; for the embedding to plot
`color_attr`	character; name of the attribute within `colData` to use for assigning colors (in lieu of `color_vec` in the `plot_embedding` function)
`color_title`	character; title to use for colors legend, defaults to the same as `color_attr`
`ellipse_attr`	character; name of the attribute within `colData` to use for drawing ellipse(s) (in lieu of `ellipse_vec` in the `plot_embedding` function)
`facet_attr`	character; name of the attribute within `colData` to use for faceting (in lieu of `facet_vec` in the `plot_embedding` function)
`...`	additional optional arguments - see `plot_embedding` function for details on other potential arguments: `xpc`, `ypc`, `plot_title`, `color_title` (if title is different from `color_attr`), `ptsize`, `saveplot`, `plotfn`, `showplot`, `returngg`, `color_pal_vec`, `dimname`

Value

default none; options to display plot (`showplot`), save plot (`saveplot`), and/or return `ggplot2` object (`returngg`)

Examples

```
library(DuoClustering2018)
library(SingleCellExperiment)
sce <- sce_full_Zhengmix4eq()[1:100,sample(1:3500,100,replace = FALSE)]
colData(sce)$Method <- matrix(sample(c('Method1','Method2'),100,replace = TRUE))
sce <- corralm(sce, splitby = 'Method')

# to plot and show only
plot_embedding_sce(sce = sce,
                   which_embedding = 'corralm',
                   xpc = 1,
                   plot_title = 'corralm: PC1 by PC2',
                   color_attr = "Method",
                   ellipse_attr = 'phenoid',
                   saveplot = FALSE)

# to return ggplot2 object and display, but not save
```

```
corralm_ggplot <- plot_embedding_sce(sce = sce,
                                     which_embedding = 'corralm',
                                     xpc = 1,
                                     plot_title = 'corralm: PC1 by PC2',
                                     color_attr = 'Method',
                                     ellipse_attr = 'phenoid',
                                     returngg = TRUE,
                                     saveplot = FALSE)
```

`rv` *rv coefficient*

Description

rv coefficient

Usage

```
rv(mat1, mat2)
```

Arguments

`mat1`	matrix (or matrix-like, e.g., df); either columns or rows should be matched with `mat2`
`mat2`	matrix (or matrix-like, e.g., df); either columns or rows should be matched with `mat1`

Value

numeric; RV coefficient between the matched matrices

Examples

```
a <- matrix(sample(1:10,100, TRUE), nrow = 10)
b <- matrix(sample(1:10,50, TRUE), nrow = 5)

rv(a, b) # matched by columns
rv(t(a), t(b)) # matched by rows
```

`scal_var` *Generate a scaled variance plot for an integrative embedding*

Description

Generate a scaled variance plot for an integrative embedding

Usage

```
scal_var(
  inp,
  batchvec = NULL,
  pcs = seq(3),
  returngg = FALSE,
  showplot = TRUE,
  plot_subtitle = NULL
)
```

Arguments

`inp`	corralm object or matrix; embedding to compute scaled variances
`batchvec`	vector; batch labels (can be numeric or char). Defaults to ‘NULL‘, which is appropriate for using a corralm object. If using an embedding matrix for inp, then this argument must be given and length must correspond to number of rows in ‘inp‘.
`pcs`	numeric; vector of which PCs should be shown. Defaults to 1:3
`returngg`	boolean; whether or not to return a `ggplot2` object, defaults `FALSE`
`showplot`	boolean; whether or not to show the plot, defaults `TRUE`
`plot_subtitle`	string; the text that should show in the subtitle for the plot. defaults to NULL

Value

N/A or a ggplot object

Examples

```
dat <- matrix(rnorm(10000), ncol = 50)
bv <- rep(seq(4),c(10,30,60,100))
scal_var(dat,bv, pcs = seq(4))
```

`scal_var_mat` *Generate a matrix of the scaled variance values*

Description

Generate a matrix of the scaled variance values

Usage

```
scal_var_mat(inp, batchvec = NULL)
```

Arguments

`inp`	corralm object or matrix; embedding to compute scaled variances
`batchvec`	vector; batch labels (can be numeric or char). Defaults to 'NULL', which is appropriate for using a corralm object. If using an embedding matrix for inp, then this argument must be given and length must correspond to number of rows in 'inp'.

Value

matrix of the scaled variance values by PC (batches in rows; PCs in columns)

Examples

```
dat <- matrix(rnorm(5000), ncol = 50)
bv <- rep(seq(3),c(10,30,60))
scal_var_mat(dat, bv)
```

`sce2matlist` *SingleCellExperiment to list of matrices*

Description

SingleCellExperiment to list of matrices

Usage

```
sce2matlist(sce, splitby, to_include = NULL, whichmat = "counts")
```

Arguments

`sce`	SingleCellExperiment that is to be separated into list of count matrices
`splitby`	character; name of the attribute from colData that should be used to separate the SCE
`to_include`	(optional) character vector; determines which values from the "splitby" column will be included in the outputted matlist. NULL is the default, and will result in selecting all elements
`whichmat`	character; defaults to `counts`, can also use `logcounts` or `normcounts` if stored in the `sce` object

Value

list of matrices

Examples

```
library(DuoClustering2018)
sce <- sce_full_Zhengmix4eq()
matlist <- sce2matlist(sce = sce, splitby = 'phenoid', whichmat = 'logcounts')
```

`trim_matdist` *Trim extreme values in a pre-processed matrix*

Description

Smooths the extreme values in a chi-square-transformed matrix to lessen the influence of "rare objects."

Usage

```
trim_matdist(mat, pct_trim = 0.01)
```

Arguments

`mat`	matrix; should be pre-processed/normalized to some sort of approximately normally distributed statistic (e.g., chi-squared transformation with ‘corral_preproc‘ or Z-score normalization)
`pct_trim`	numeric; the percent of observations to smooth. Defaults to ‘pct_trim‘ = .01, which corresponds to smoothing all observations to be between the .5 percentile and 99.5 percentile range of the input matrix

Details

(Usually not called directly; can be included by using the ‘smooth‘ argument in the ‘corral‘, ‘corralm‘, and ‘corral_preproc‘ functions)

Value

smoothed matrix

Examples

```
count_mat <- matrix(rpois(10000, 300)*rbinom(10000,1,.1), ncol = 100)
smoothed_preproc_mat <- corral_preproc(count_mat, smooth = TRUE)
```

`var_stabilize` *Apply a variance stabilizing transformation*

Description

Prior to running CA, there is an option to apply a variance stabilizing transformation. This function can be called explicitly or used with the ‘vst_mth‘ argument in `corral` and `corral_preproc`.

Usage

```
var_stabilize(inp, transform = c("sqrt", "freemantukey", "anscombe"))
```

Arguments

`inp`	matrix, numeric, counts or logcounts; can be sparse Matrix or matrix
`transform`	character indicating which method should be applied. Defaults to the square root transform (‘"sqrt"‘). Other options include ‘"freemantukey"‘ and ‘"anscombe"‘.

Value

variance-stabilized matrix; sparse if possible

Examples

```
x <- as.matrix(rpois(100, lambda = 50), ncol = 10)
vst_x <- var_stabilize(x)
```

Index

∗ **internal**
    obs2probs, 15

add_embeddings2scelist, 2
all_are, 3

biplot_corral, 4

compsvd, 5
corral, *6*, *9–11*
corral *(*corral_mat*)*, 8
corral_mat, *8*, 8, *9*, *10*, *13*
corral_preproc, *5*, 10
corral_sce, *8*
corral_sce *(*corral_mat*)*, 8
corralm, *2*, *6*, *9–11*
corralm *(*corralm_matlist*)*, 6
corralm_matlist, 6, *7*, *13*
corralm_sce, *7*
corralm_sce *(*corralm_matlist*)*, 6

earthmover_dist, 12

get_pct_var_exp_svd, 12
get_weights, 13
ggplot2, *17*, *18*, *20*

list2mat, 14

na2zero, 14

obs2probs, 15

pairwise_rv, 15
plot_embedding, 16, *18*
plot_embedding_sce, 17
print.corral *(*corral_mat*)*, 8
print.corralm *(*corralm_matlist*)*, 6

rv, 19

scal_var, 20
scal_var_mat, 21
sce2matlist, *7*, 21
SingleCellExperiment, *8*, *10*, *18*

trim_matdist, 22

var_stabilize, 23