

Package ‘veloviz’

December 20, 2025

Title VeloViz: RNA-velocity informed 2D embeddings for visualizing cell state trajectories

Version 1.17.0

Description VeloViz uses each cell’s current observed and predicted future transcriptional states inferred from RNA velocity analysis to build a nearest neighbor graph between cells in the population. Edges are then pruned based on a cosine correlation threshold and/or a distance threshold and the resulting graph is visualized using a force-directed graph layout algorithm. VeloViz can help ensure that relationships between cell states are reflected in the 2D embedding, allowing for more reliable representation of underlying cellular trajectories.

biocViews Transcriptomics, Visualization, GeneExpression, Sequencing, RNASeq, DimensionReduction

License GPL-3

Encoding UTF-8

LazyData false

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

Imports Rcpp, Matrix, igraph, mgcv, RSpectra, grDevices, graphics, stats

LinkingTo Rcpp

Depends R (>= 4.1)

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

git_url <https://git.bioconductor.org/packages/veloviz>

git_branch devel

git_last_commit 3dd61ba

git_last_commit_date 2025-10-29

Repository Bioconductor 3.23

Date/Publication 2025-12-19

Author Lyla Atta [aut, cre] (ORCID: <<https://orcid.org/0000-0002-6113-0082>>),
Jean Fan [aut] (ORCID: <<https://orcid.org/0000-0002-0212-5451>>),
Arpan Sahoo [aut] (ORCID: <<https://orcid.org/0000-0002-0325-2073>>)

Maintainer Lyla Atta <lylaatta@jhmi.edu>

Contents

asNNGraph	2
buildVeloviz	3
graphViz	5
normalizeDepth	7
normalizeVariance	7
pancreas	9
plotEmbedding	9
plotVeloviz	11
projectedNeighbors	12
reduceDimensions	13
vel	14
veloviz	15
Index	16

asNNGraph	<i>Function to produce idx and dist representation of a VeloViz graph</i>
-----------	---

Description

Function to produce idx and dist representation of a VeloViz graph

Usage

asNNGraph(vig)

Arguments

vig output of buildVeloviz

Value

idx numVertices x numNeighbors matrix, where each row i contains indices of vertex i’s neighbors
dist numVertices x numNeighbors matrix, where each row i contains distances from vertex i to its neighbors

Examples

```
data(vel)
curr <- vel$current
proj <- vel$projected

vv <- buildVeloviz(curr = curr, proj = proj, normalize.depth = TRUE,
  use.ods.genes = FALSE, alpha = 0.05, pca = TRUE, nPCs = 3, center = TRUE,
  scale = TRUE, k = 10, similarity.threshold = -1, distance.weight = 1,
  distance.threshold = 1, weighted = TRUE, verbose = FALSE)

asNNGraph(vv)
```

buildVeloviz	<i>Creates VeloViz graph and FDG layout from PC scores of current and projected transcriptional states.</i>
--------------	---

Description

Creates VeloViz graph and FDG layout from PC scores of current and projected transcriptional states.

Usage

```
buildVeloviz(
  curr,
  proj,
  normalize.depth = TRUE,
  depth = 1e+06,
  use.ods.genes = TRUE,
  max.ods.genes = 2000,
  alpha = 0.05,
  pca = TRUE,
  center = TRUE,
  scale = TRUE,
  nPCs = 10,
  k = 10,
  similarity.threshold = 0,
  distance.weight = 1,
  distance.threshold = 1,
  weighted = TRUE,
  remove.unconnected = TRUE,
  verbose = FALSE,
  details = FALSE
)
```

Arguments

<code>curr</code>	Genes (rows) x cells (columns) matrix of observed current transcriptional state
<code>proj</code>	Genes (rows) x cells (columns) matrix of predicted future transcriptional state
<code>normalize.depth</code>	logical to normalize raw counts to counts per million, default = TRUE
<code>depth</code>	Depth scaling, default = 1e6 for counts per million (CPM)
<code>use.ods.genes</code>	Use only overdispersed genes to create VeloViz graph, default = TRUE
<code>max.ods.genes</code>	number of most highly expressed overdispersed genes to use to create VeloViz graph, default = 2000
<code>alpha</code>	Significance threshold for overdispersed genes, default = 0.05
<code>pca</code>	logical to use PC scores to create VeloViz graph, default = TRUE. FALSE = use gene expression to create VeloViz graph
<code>center</code>	logical to mean center gene expression before PCA, default = TRUE
<code>scale</code>	logical to scale gene expression variance before PCA, default = TRUE
<code>nPCs</code>	number of principal components to use to create VeloViz graph, default = 10
<code>k</code>	Number of nearest neighbors to assign each cell
<code>similarity.threshold</code>	similarity threshold below which to remove edges, default = -1 i.e. no edges removed
<code>distance.weight</code>	Weight of distance component of composite distance, default = 1
<code>distance.threshold</code>	quantile threshold for distance component above which to remove edges, default = 1 i.e. no edges removed
<code>weighted</code>	logical indicating whether to compute VeloViz edges based on composite distance, default = TRUE. FALSE = all edges are of equal weight
<code>remove.unconnected</code>	logical indicating whether to remove cells with no edges in the VeloViz graph from the output embedding, default = TRUE (removed)
<code>verbose</code>	logical for verbosity setting, default = FALSE
<code>details</code>	logical to return detailed data frame or names of genes, default = FALSE

Value

`graph` igraph object of VeloViz graph
`fdg_coords` cells (rows) x 2 coordinates of force-directed layout of VeloViz graph
`projectedNeighbors` output of `projectedNeighbors`

See Also

[projectedNeighbors](#)

Examples

```
data(vel)
curr <- vel$current
proj <- vel$projected

buildVeloviz(curr = curr, proj = proj, normalize.depth = TRUE,
use.ods.genes = TRUE, alpha = 0.05, pca = TRUE, nPCs = 20, center = TRUE,
scale = TRUE, k = 5, similarity.threshold = 0.25, distance.weight = 1,
distance.threshold = 0.5, weighted = TRUE, verbose = FALSE)
```

graphViz

*Visualize as velocity informed force directed graph***Description**

Visualize as velocity informed force directed graph

Usage

```
graphViz(
  observed,
  projected,
  k,
  distance_metric = "L2",
  similarity_metric = "cosine",
  distance_weight = 1,
  distance_threshold = 1,
  similarity_threshold = -1,
  weighted = TRUE,
  remove_unconnected = TRUE,
  return_graph = FALSE,
  plot = TRUE,
  cell.colors = NA,
  title = NA
)
```

Arguments

observed	PCs (rows) x cells (columns) matrix of observed transcriptional state projected into PC space
projected	PCs (rows) x cells (columns) matrix of projected transcriptional states. Cell should be in same order as in observed
k	Number of nearest neighbors to assign each cell
distance_metric	Method to compute distance component of composite distance. "L1" or "L2", default = "L2"

similarity_metric	Method to compute similarity between velocity and cell transition matrices. "cosine" or "pearson", default = "cosine"
distance_weight	Weight of distance component of composite distance, default = 1
distance_threshold	quantile threshold for distance component above which to remove edges, default = 1 i.e. no edges removed
similarity_threshold	similarity threshold below which to remove edges, default = -1 i.e. no edges removed
weighted	if TRUE, assigns edge weights based on composite distance, if FALSE assigns equal weights to all edges, default = TRUE
remove_unconnected	if TRUE, does not plot cells with no edges, default = TRUE
return_graph	if TRUE, returns igraph object graph, force-directed layout coordinates fdg_coords, and projected_neighbors object detailing composite distance values and components, default = FALSE
plot	if TRUE, plots graph and force-directed layout
cell.colors	cell.colors to use for plotting
title	title to use for plot

Value

graph igraph object of VeloViz graph
 fdg_coords cells (rows) x 2 coordinates of force-directed layout of VeloViz graph
 projectedNeighbors output of projectedNeighbors

See Also

[projectedNeighbors](#)

Examples

```
data(vel)
curr = vel$current
proj = vel$projected

m <- log10(curr+1)
pca <- RSpectra::svds(A = Matrix::t(m), k=3,
  opts = list(center = FALSE, scale = FALSE, maxitr = 2000, tol = 1e-10))
pca.curr <- Matrix::t(m) %*% pca$v[,1:3]

m <- log10(proj+1)
pca.proj <- Matrix::t(m) %*% pca$v[,1:3]

graphViz(t(pca.curr), t(pca.proj), k=10,
```

```
cell.colors=NA, similarity_threshold=-1, distance_weight = 1,  
distance_threshold = 1, weighted = TRUE, remove_unconnected = TRUE,  
plot = FALSE, return_graph = TRUE)
```

normalizeDepth	<i>Normalizes counts to CPM</i>
----------------	---------------------------------

Description

Normalizes raw counts to counts per million

Usage

```
normalizeDepth(counts, depthScale = 1e+06, verbose = TRUE)
```

Arguments

counts	Read count matrix. The rows correspond to genes, columns correspond to individual cells
depthScale	Depth scaling. Using a million for CPM (default: 1e6)
verbose	Boolean for verbosity setting (default: TRUE)

Value

a normalized matrix

Examples

```
data(vel)  
curr <- vel$current  
  
normalizeDepth(curr)
```

normalizeVariance	<i>Identify overdispersed genes by normalizing counts per million (CPM) gene expression variance relative to transcriptome-wide expectations (Modified from SCDE/PAGODA2 code)</i>
-------------------	--

Description

Normalizes gene expression magnitudes to with respect to its ratio to the transcriptome-wide expectation as determined by local regression on expression magnitude

Usage

```
normalizeVariance(  
  cpm,  
  gam.k = 5,  
  alpha = 0.05,  
  max.adjusted.variance = 1000,  
  min.adjusted.variance = 0.001,  
  verbose = TRUE,  
  plot = FALSE,  
  details = FALSE  
)
```

Arguments

cpm	Counts per million (CPM) matrix. Rows are genes, columns are cells.
gam.k	Generalized additive model parameter; the dimension of the basis used to represent the smooth term (default: 5)
alpha	Significance threshold for overdispersed genes (default: 0.05)
max.adjusted.variance	Ceiling on maximum variance after normalization to prevent infinities (default: 1e3)
min.adjusted.variance	Floor on minimum variance after normalization (default: 1e-3)
verbose	Boolean for verbosity setting (default: TRUE)
plot	Boolean to plot mean variance plots before and after correction
details	Boolean to return detailed data frame or names of genes (default: FALSE)

Value

A list with two items: (1) an adjusted CPM matrix with the same dimensions as the input and (2) a dataframe with the summary statistics for each gene.

Examples

```
data(vel)  
curr <- vel$current  
  
normalizeDepth(curr)
```

pancreas	<i>Pancreas scRNA-seq data</i>
----------	--------------------------------

Description

Pancreatic endocrinogenesis scRNA-seq from Bastidas-Ponce et. al., Development 2019 accessed via scVelo package and subsampled to 739 cells.

Usage

pancreas

Format

list of 4 objects:

spliced matrix, 7192 genes x 739 cells of spliced counts

unspliced matrix, 7192 genes x 739 cells of unspliced counts

pcs matrix, 739 x 50 cell scores in 50 PCs

clusters factor of cell type annotations from scVelo

Source

<https://dev.biologists.org/content/146/12/dev173849.long>

plotEmbedding	<i>Plot 2D embedding From scde/pagoda2/MUDAN</i>
---------------	--

Description

Plot 2D embedding From scde/pagoda2/MUDAN

Usage

```
plotEmbedding(  
  emb,  
  groups = NULL,  
  colors = NULL,  
  cex = 0.6,  
  alpha = 0.4,  
  gradientPalette = NULL,  
  zlim = NULL,  
  s = 1,  
  v = 0.8,  
  min.group.size = 1,
```

```

    show.legend = FALSE,
    mark.clusters = FALSE,
    mark.cluster.cex = 2,
    shuffle.colors = FALSE,
    legend.x = "topright",
    gradient.range.quantile = 0.95,
    verbose = TRUE,
    unclassified.cell.color = "gray70",
    group.level.colors = NULL,
    ...
)

```

Arguments

<code>emb</code>	dataframe with x and y coordinates
<code>groups</code>	factor annotations for rows on <code>emb</code> for visualizing cluster annotations
<code>colors</code>	color or numeric values for rows on <code>emb</code> for visualizing gene expression
<code>cex</code>	point size
<code>alpha</code>	point opacity
<code>gradientPalette</code>	palette for colors if numeric values provided
<code>zlim</code>	range for colors
<code>s</code>	saturation of rainbow for group colors
<code>v</code>	value of rainbow for group colors
<code>min.group.size</code>	minimum size of group in order for group to be colored
<code>show.legend</code>	whether to show legend
<code>mark.clusters</code>	whether to mark clusters with name of cluster
<code>mark.cluster.cex</code>	cluster marker point size
<code>shuffle.colors</code>	whether to shuffle group colors
<code>legend.x</code>	legend position ie. 'topright', 'topleft', 'bottomleft', 'bottomright'
<code>gradient.range.quantile</code>	quantile for mapping colors to gradient palette
<code>verbose</code>	verbosity
<code>unclassified.cell.color</code>	cells not included in groups will be labeled in this color
<code>group.level.colors</code>	set group level colors. Default uses rainbow.
<code>...</code>	Additional parameters to pass to <code>BASE::plot</code>

Value

embedding plot

Examples

```

data(vel)
curr <- vel$current
proj <- vel$projected

vv <- buildVeloviz(curr = curr, proj = proj, normalize.depth = TRUE,
  use.ods.genes = TRUE, alpha = 0.05, pca = TRUE, nPCs = 20, center = TRUE,
  scale = TRUE, k = 5, similarity.threshold = 0.25, distance.weight = 1,
  distance.threshold = 0.5, weighted = TRUE, verbose = FALSE)

plotEmbedding(vv$fdg_coords)

```

plotVeloviz	<i>Plot function</i>
-------------	----------------------

Description

Plot function

Usage

```

plotVeloviz(
  vig,
  layout.method = igraph::layout_with_fr,
  clusters = NA,
  cluster.method = igraph::cluster_louvain,
  col = NA,
  alpha = 0.05,
  verbose = TRUE
)

```

Arguments

vig	output of buildVeloviz
layout.method	igraph method to use for generating 2D graph representation, default = igraph::layout_with_fr
clusters	cluster annotations for cells in data
cluster.method	igraph method to use for clustering if clusters are not provided, default = igraph::cluster_louvain
col	colors to use for plotting
alpha	transparency for plotting graph nodes
verbose	logical for verbosity setting, default = FALSE

Value

cells (rows) x 2 coordinates of force-directed layout of VeloViz graph

Examples

```

data(vel)
curr <- vel$current
proj <- vel$projected

vv <- buildVeloviz(curr = curr, proj = proj, normalize.depth = TRUE,
  use.ods.genes = TRUE, alpha = 0.05, pca = TRUE, nPCs = 20, center = TRUE,
  scale = TRUE, k = 5, similarity.threshold = 0.25, distance.weight = 1,
  distance.threshold = 0.5, weighted = TRUE, verbose = FALSE)

plotVeloviz(vv)

```

projectedNeighbors	<i>Computes composite distances between all cell pairs and returns k-nearest neighbors and edge weights needed to build VeloViz graph.</i>
--------------------	--

Description

Computes composite distances between all cell pairs and returns k-nearest neighbors and edge weights needed to build VeloViz graph.

Usage

```

projectedNeighbors(
  observed,
  projected,
  k,
  distance_metric = "L2",
  similarity_metric = "cosine",
  distance_weight = 1,
  distance_threshold = 1,
  similarity_threshold = -1
)

```

Arguments

observed	PCs (rows) x cells (columns) matrix of observed transcriptional state projected into PC space
projected	PCs (rows) x cells (columns) matrix of projected transcriptional states. Cells should be in same order as in observed
k	Number of nearest neighbors to assign each cell
distance_metric	Method to compute distance component of composite distance. "L1" or "L2", default = "L2"

`similarity_metric`
 Method to compute similarity between velocity and cell transition matrices. "cosine" or "pearson", default = "cosine"

`distance_weight`
 Weight of distance component of composite distance, default = 1

`distance_threshold`
 quantile threshold for distance component above which to remove edges, default = 1 i.e. no edges removed

`similarity_threshold`
 similarity threshold below which to remove edges, default = -1 i.e. no edges removed

Value

`kNNs` cells (rows) x `k` (columns) matrix of indices of each cell's nearest neighbors computed based on composite distance. Edges removed based on distance or similarity threshold will be NA.

`edge_weights` cells (rows) x `k` (columns) matrix of edge weights computed based on composite distance. Edges removed based on distance or similarity threshold will be NA.

`all_dists` cells x cells matrix of all pairwise composite distances

`dist_comp` components of composite distance: `invDist` distance component, `negSim` similarity component

See Also

[graphViz](#)

Examples

```
data(vel)
curr <- vel$current
proj <- vel$projected

projectedNeighbors(curr, proj, 10)
```

<code>reduceDimensions</code>	<i>Reduce dimension using Principal Components Analysis via svds from RSpectra</i>
-------------------------------	--

Description

Reduce dimension using Principal Components Analysis via svds from RSpectra

Usage

```
reduceDimensions(
  matnorm,
  center = TRUE,
  scale = TRUE,
  max.ods.genes = 2000,
  nPCs = 50,
  verbose = TRUE,
  plot = FALSE,
  details = FALSE
)
```

Arguments

matnorm	matrix on which to perform PCA
center	logical to mean center gene expression before PCA, default = TRUE
scale	logical to scale gene expression variance before PCA, default = TRUE
max.ods.genes	number of most highly expressed overdispersed genes to include, default = 2000
nPCs	number of principal components to reduce to return, default = 50
verbose	logical for verbosity setting, default = TRUE
plot	plot singular values vs number of components
details	logical to return pca object, default = FALSE

Value

matrix of cell scores in nPCs components

Examples

```
data(vel)
curr <- vel$current

curr.norm <- normalizeDepth(curr)
curr.norm <- log10(curr.norm+1)
reduceDimensions(curr.norm, nPCs=3)
```

vel

MERFISH velocity subset

Description

output of `velocyto.R::gene.relative.velocity.estimates` for 40 cell subset of MERFISH data. Used to run examples

Usage

vel

Format

list of 1:

vel velocity output containing current observed ("current") and predicted future ("projected") estimates

Source

<https://www.pnas.org/content/116/39/19490>

veloviz	<i>veloviz</i>
---------	----------------

Description

Package for creating RNA velocity informed embeddings for single cell transcriptomics

Index

* datasets

pancreas, [9](#)

vel, [14](#)

asNNGraph, [2](#)

buildVeloviz, [3](#)

graphViz, [5](#), [13](#)

normalizeDepth, [7](#)

normalizeVariance, [7](#)

pancreas, [9](#)

plotEmbedding, [9](#)

plotVeloviz, [11](#)

projectedNeighbors, [4](#), [6](#), [12](#)

reduceDimensions, [13](#)

vel, [14](#)

veloviz, [15](#)