Package: MAGICR (via r-universe)

May 25, 2026

Type Package

Title Markov Affinity-Based Graph Imputation of Cells

Version 1.0.0

Date 2026-01-26

Author Zaoqu Liu [aut, cre] (<https://orcid.org/0000-0002-0452-742X>), Krishnaswamy Lab [cph] (Original MAGIC algorithm)

Maintainer Zaoqu Liu <liuzaoqu@163.com>

Description Native R implementation of MAGIC (Markov Affinity-based Graph Imputation of Cells) for denoising and imputation of single-cell RNA sequencing data. MAGIC uses diffusion geometry to denoise single-cell data and fill in missing transcripts, as described in van Dijk et al. (2018) <doi:10.1016/j.cell.2018.05.061>. This package provides a pure R implementation with optional C++ acceleration, parallel computing support, and seamless integration with Seurat (v4 and v5) objects.

License GPL-2

URL https://zaoqu-liu.github.io/MAGICR, https://github.com/Zaoqu-Liu/MAGICR

BugReports https://github.com/Zaoqu-Liu/MAGICR/issues

Encoding UTF-8

LazyData true

Depends R (>= 3.6.0)

Imports Matrix (>= 1.3-0), irlba (>= 2.3.0), RANN (>= 2.6.0), Rcpp, rlang, methods, stats, utils, future, grDevices, graphics

Suggests Seurat (>= 4.0.0), SeuratObject, SingleCellExperiment, SummarizedExperiment, S4Vectors, ggplot2, gifski, gridExtra, testthat (>= 3.0.0), knitr, rmarkdown

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 7.3.3

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation yes

Biarch true

Repository https://zaoqu-liu.r-universe.dev

Date/Publication 2026-01-25 17:58:56 UTC

RemoteUrl https://github.com/Zaoqu-Liu/MAGICR

RemoteRef main

RemoteSha c6c90d3e0cb77027aa07901d4eb6355ca898bb83

Contents

[.magic *Subset MAGIC result*

Description

Subset MAGIC result

Usage

```
## S3 method for class 'magic'
x[i, j, ...]
```

Arguments

x	MAGIC object
i	Row indices
j	Column indices
...	Additional arguments

animate_magic *Animate MAGIC Diffusion*

Description

Create an animation showing how gene-gene relationships evolve with increasing diffusion time t.

Usage

```
animate_magic(
  data,
  gene_x,
  gene_y,
  gene_color = NULL,
  t_max = 20,
  delay = 2,
  operator = NULL,
  filename = NULL,
  width = 600,
  height = 500,
  point_size = 1,
  interval = 200,
  verbose = FALSE,
  ...
)
```

Arguments

data	Input data matrix (cells x genes).
gene_x	Gene for x-axis (name or index).
gene_y	Gene for y-axis (name or index).
gene_color	Gene for coloring points (optional).
t_max	Maximum t value. Default is 20.
delay	Number of initial frames before diffusion. Default is 2.
operator	Pre-computed magic object. Default is NULL.
filename	Output filename (.gif). Default is NULL (no save).
width	Plot width in pixels. Default is 600.
height	Plot height in pixels. Default is 500.
point_size	Point size. Default is 1.
interval	Milliseconds between frames. Default is 200.
verbose	Verbosity. Default is FALSE.
...	Additional arguments passed to magic().

Details

This function creates an animation showing how MAGIC diffusion progressively reveals gene-gene relationships. It requires the `gganimate` and `gifski` packages for GIF output.

Value

A list of ggplot objects (one per frame), or writes a GIF file.

Examples

```
## Not run:
data(magic_testdata)

# Create animation frames
frames <- animate_magic(magic_testdata,
                        gene_x = 1, gene_y = 2,
                        t_max = 10)

# Save as GIF (requires gganimate and gifski)
animate_magic(magic_testdata,
              gene_x = 1, gene_y = 2,
              t_max = 10,
              filename = "magic_animation.gif")

## End(Not run)
```

compute_alpha_kernel *Compute Alpha-Decaying Kernel*

Description

Computes the alpha-decaying kernel from kNN distances.

Usage

```
compute_alpha_kernel(knn_graph, decay = 1, threshold = 1e-04, verbose = FALSE)
```

Arguments

knn_graph	Result from magic_knn_graph.
decay	Numeric. Alpha parameter. If NULL, uses unweighted kernel. Default is 1.
threshold	Numeric. Values below this are set to 0. Default is 1e-4.
verbose	Logical or integer. Verbosity level.

Details

The alpha-decaying kernel:

$$K(i,j) = \exp\left(-\left(\frac{d(i,j)}{\epsilon_i}\right)^{\alpha}\right)$$

where ϵ_i is the distance to the k-th neighbor (adaptive bandwidth) and α controls the decay rate.

Reference: Moon, K.R., et al. (2019). Visualizing structure and transitions in high-dimensional biological data. Nature Biotechnology, 37(12), 1482-1492.

Value

A sparse symmetric kernel matrix (dgCMatrix).

diffusion *Diffusion Operations*

Description

Functions for diffusion-based data imputation.

dremi *kNN-DREMI: Conditional Density Resampled Mutual Information*

Description

Functions for computing kNN-DREMI on single-cell data.

GetMAGICData *Extract MAGIC Results from Seurat Object*

Description

Convenience function to extract MAGIC-imputed data from a Seurat object.

Usage

```
GetMAGICData(
  object,
  assay = "MAGIC_RNA",
  genes = NULL,
  cells = NULL,
  as_matrix = TRUE
)
```

Arguments

object	Seurat object that has been processed with magic()
assay	Name of the MAGIC assay. Default is "MAGIC_RNA".
genes	Genes to extract. Default is NULL (all genes).
cells	Cells to extract. Default is NULL (all cells).
as_matrix	Logical. Return as matrix instead of data.frame? Default TRUE.

Value

Matrix or data.frame of MAGIC-imputed expression values (cells x genes).

Examples

```
## Not run:
# After running magic() on Seurat object
imputed_data <- GetMAGICData(seurat_obj)
imputed_subset <- GetMAGICData(seurat_obj, genes = c("Gene1", "Gene2"))

## End(Not run)
```

has_cpp_acceleration *Check if C++ Acceleration is Available*

Description

Returns TRUE if C++ functions compiled successfully.

Usage

```
has_cpp_acceleration()
```

Value

Logical

Examples

```
has_cpp_acceleration()
```

kernel *Kernel and Graph Construction*

Description

Functions for kNN graph and diffusion kernel construction.

knnDREMI *Compute kNN-DREMI*

Description

Calculates k-Nearest Neighbor conditional Density Resampled Estimate of Mutual Information as defined in van Dijk et al, 2018.

Usage

```
knnDREMI(
  x,
  y,
  k = 10,
  n_bins = 20,
  n_mesh = 3,
  n_jobs = 1,
  plot = FALSE,
  return_drevi = FALSE,
  ...
)
```

Arguments

x	Numeric vector. Independent variable (gene X).
y	Numeric vector. Dependent variable (gene Y).
k	Integer. Number of nearest neighbors. Default is 10.
n_bins	Integer. Number of bins for density resampling. Default is 20.
n_mesh	Integer. Mesh density within bins. Default is 3.
n_jobs	Integer. Number of parallel workers (not used in R). Default is 1.
plot	Logical. Create diagnostic plots. Default is FALSE.
return_drevi	Logical. Also return DREVI density matrix. Default is FALSE.
...	Additional arguments passed to plot function.

Details

kNN-DREMI is an adaptation of DREMI (Krishnaswamy et al. 2014) for single cell RNA-sequencing data. DREMI captures the functional relationship between two genes across their entire dynamic range. The key change to kNN-DREMI is the replacement of the heat diffusion-based kernel-density estimator by a k-nearest neighbor-based density estimator.

Note: kNN-DREMI, like Mutual Information and DREMI, is not symmetric. Here we are estimating I(Y|X).

This implementation follows exactly the scprep Python implementation.

Value

If return_drevi is FALSE, returns the DREMI value (numeric). If return_drevi is TRUE, returns a list with dremi and drevi (density matrix).

References

van Dijk, D., et al. (2018). Recovering gene interactions from single-cell data using data diffusion. Cell, 174(3), 716-729. doi:10.1016/j.cell.2018.05.061

Krishnaswamy, S., et al. (2014). Conditional density-based analysis of T cell signaling in single-cell data. Science, 346(6213), 1250689.

Examples

```
## Not run:
# After running MAGIC
result <- magic(data, t = 3)
imputed <- as.matrix(result)

# Compute kNN-DREMI between two genes
dremi <- knnDREMI(imputed[, "GeneA"], imputed[, "GeneB"])

# With diagnostic plot
dremi <- knnDREMI(imputed[, "GeneA"], imputed[, "GeneB"], plot = TRUE)

## End(Not run)
```

library_size_normalize *Library Size Normalization*

Description

Normalizes data by library size (total counts per cell).

Usage

```
library_size_normalize(data, verbose = FALSE)
```

Arguments

data	Matrix (cells x genes).
verbose	Logical. Print progress. Default is FALSE.

Details

Formula: $X_{norm} = X / rowSums(X) * median(rowSums(X))$

Value

Normalized matrix.

log_transform *Log Transformation*

Description

Applies log transformation with pseudo-count.

Usage

```
log_transform(data, pseudo_count = 1, base = exp(1), verbose = FALSE)
```

Arguments

data	Matrix.
pseudo_count	Numeric. Default is 1.
base	Numeric. Log base. Default is exp(1).
verbose	Logical. Default is FALSE.

Details

Formula: $X_{log} = log_{base}(X + pseudo\_count)$

Value

Transformed matrix.

`magic` *MAGIC: Markov Affinity-based Graph Imputation of Cells*

Description

Main function for MAGIC imputation of single-cell data.

Performs MAGIC (Markov Affinity-based Graph Imputation of Cells) for denoising and imputation of single-cell data.

Run MAGIC on a Seurat object and store results in a new assay.

Run MAGIC on a SingleCellExperiment object.

Usage

```
magic(data, ...)

## Default S3 method:
magic(
  data,
  genes = NULL,
  knn = 5,
  knn_max = NULL,
  decay = 1,
  t = 3,
  npca = 100,
  solver = c("exact", "approximate"),
  init = NULL,
  t_max = 20,
  knn_dist = c("euclidean", "cosine", "manhattan", "correlation"),
  n_jobs = 1,
  seed = NULL,
  verbose = TRUE,
  ...
)

## S3 method for class 'matrix'
magic(
  data,
  genes = NULL,
  knn = 5,
  knn_max = NULL,
  decay = 1,
  t = 3,
```

```
  npca = 100,
  solver = c("exact", "approximate"),
  init = NULL,
  t_max = 20,
  knn_dist = c("euclidean", "cosine", "manhattan", "correlation"),
  n_jobs = 1,
  seed = NULL,
  verbose = TRUE,
  ...
)

## S3 method for class 'dgCMatrix'
magic(data, ...)

## S3 method for class 'data.frame'
magic(data, ...)

## S3 method for class 'Seurat'
magic(
  data,
  assay = NULL,
  slot = "data",
  genes = "all_genes",
  new_assay_name = NULL,
  knn = 5,
  knn_max = NULL,
  decay = 1,
  t = 3,
  npca = 100,
  solver = c("exact", "approximate"),
  init = NULL,
  t_max = 20,
  knn_dist = c("euclidean", "cosine", "manhattan", "correlation"),
  n_jobs = 1,
  seed = NULL,
  verbose = TRUE,
  ...
)

## S3 method for class 'SingleCellExperiment'
magic(
  data,
  assay_name = "logcounts",
  genes = "all_genes",
  new_assay_name = "MAGIC",
  knn = 5,
  knn_max = NULL,
  decay = 1,
```

```
  t = 3,
  npca = 100,
  solver = c("exact", "approximate"),
  init = NULL,
  t_max = 20,
  knn_dist = c("euclidean", "cosine", "manhattan", "correlation"),
  n_jobs = 1,
  seed = NULL,
  verbose = TRUE,
  ...
)
```

Arguments

data	A SingleCellExperiment object.
...	Additional arguments.
genes	Gene selection. Default is "all_genes".
knn	Number of neighbors. Default is 5.
knn_max	Maximum neighbors. Default is NULL.
decay	Alpha decay. Default is 1.
t	Diffusion steps. Default is 3.
npca	PCA components. Default is 100.
solver	Solver type. Default is "exact".
init	Previous result. Default is NULL.
t_max	Maximum t. Default is 20.
knn_dist	Distance metric. Default is "euclidean".
n_jobs	Parallel workers. Default is 1.
seed	Random seed. Default is NULL.
verbose	Verbosity. Default is TRUE.
assay	Character. Name of the assay to use as input. Default is the default assay of the object.
slot	Character. Slot to use for input data. Default is "data" (normalized data). For Seurat v5, this corresponds to the layer name.
new_assay_name	Name for the new assay. Default is "MAGIC".
assay_name	Character. Name of the assay to use. Default is "logcounts".

Details

MAGIC denoises single-cell data using diffusion on a cell similarity graph:

1. Optional PCA dimension reduction
2. Build k-nearest neighbor graph
3. Construct alpha-decaying kernel

4. Create row-stochastic diffusion operator
5. Apply diffusion operator t times

The imputation formula is: $X_{imputed} = P^t X$

This function:

1. Extracts normalized data from the specified assay
2. Transposes to cells x genes format
3. Runs MAGIC imputation
4. Creates a new assay with the imputed data
5. Stores MAGIC parameters in the object's misc slot

The function is compatible with both Seurat v4 and v5. For v4, it uses GetAssayData/SetAssayData. For v5, it uses LayerData where available.

Value

A "magic" object (for matrices) or modified input object (for Seurat/SCE).

The Seurat object with a new assay containing MAGIC-imputed values. The new assay is named "MAGIC_<original_assay>" by default.

SingleCellExperiment object with MAGIC results in a new assay.

References

van Dijk, D., et al. (2018). Recovering gene interactions from single-cell data using data diffusion. Cell, 174(3), 716-729. doi:10.1016/j.cell.2018.05.061

Examples

```
## Not run:
# Basic usage
set.seed(42)
data <- matrix(rpois(5000, 2), nrow = 500, ncol = 10)
result <- magic(data, t = 3)
imputed <- as.matrix(result)

# Automatic t selection
result <- magic(data, t = "auto")

# Specific genes
result <- magic(data, genes = c(1, 2, 3))

## End(Not run)
## Not run:
library(Seurat)

# Create example Seurat object
counts <- matrix(rpois(3000, lambda = 5), nrow = 30, ncol = 100)
rownames(counts) <- paste0("Gene", 1:30)
```

```
colnames(counts) <- paste0("Cell", 1:100)

seurat_obj <- CreateSeuratObject(counts = counts)
seurat_obj <- NormalizeData(seurat_obj)

# Run MAGIC
seurat_obj <- magic(seurat_obj, t = 3)

# Access MAGIC results
GetAssayData(seurat_obj, assay = "MAGIC_RNA", slot = "data")

# Make MAGIC assay the default
DefaultAssay(seurat_obj) <- "MAGIC_RNA"

## End(Not run)
```

`magic_diffusion_operator` *Compute Diffusion Operator*

Description

Creates the row-stochastic diffusion operator (Markov matrix).

Usage

```
magic_diffusion_operator(kernel, verbose = FALSE)
```

Arguments

`kernel`	Symmetric kernel matrix from `compute_alpha_kernel`.
`verbose`	Logical or integer. Verbosity level.

Details

The diffusion operator is:

$$P = D^{-1}K$$

where D is the diagonal matrix of row sums.

Value

A row-stochastic sparse matrix (each row sums to 1).

`magic_impute` *Perform MAGIC Imputation*

Description

Applies the diffusion operator to impute and denoise data.

Usage

```
magic_impute(
  data,
  diff_op,
  t = 3,
  t_max = 20,
  threshold = 0.001,
  solver = c("exact", "approximate"),
  verbose = FALSE
)
```

Arguments

`data`	Matrix (cells x genes) to impute.
`diff_op`	Diffusion operator (Markov matrix).
`t`	Integer or "auto". Number of diffusion steps.
`t_max`	Integer. Maximum t for automatic selection. Default is 20.
`threshold`	Numeric. Convergence threshold for automatic t. Default is 0.001.
`solver`	Character. "exact" or "approximate". Default is "exact".
`verbose`	Logical or integer. Verbosity level.

Details

MAGIC imputation applies the diffusion operator repeatedly:

$$X_{imputed} = P^t \times X$$

When t = "auto", the algorithm stops when the Procrustes disparity between consecutive iterations falls below the threshold.

Value

A matrix of imputed values.

magic_knn_graph *Build k-Nearest Neighbors Graph*

Description

Constructs a kNN graph from input data.

Usage

```
magic_knn_graph(
  data,
  knn = 5,
  knn_max = NULL,
  distance = c("euclidean", "cosine", "manhattan", "correlation"),
  verbose = FALSE
)
```

Arguments

data	Matrix (cells x features).
knn	Integer. Number of neighbors for bandwidth. Default is 5.
knn_max	Integer or NULL. Maximum neighbors. Default is 3 * knn.
distance	Character. Distance metric: "euclidean", "cosine", "manhattan", or "correlation". Default is "euclidean".
verbose	Logical or integer. Verbosity level.

Value

A list with idx (neighbor indices), dist (distances), and parameters.

magic_knnDREMI *Compute kNN-DREMI on MAGIC result*

Description

Convenience function to compute kNN-DREMI directly from a MAGIC result object.

Usage

```
magic_knnDREMI(magic_result, gene_x, gene_y, ...)
```

Arguments

magic_result	A magic object from magic().
gene_x	Name or index of gene X.
gene_y	Name or index of gene Y.
...	Additional arguments passed to knnDREMI.

Value

DREMI value or list with DREMI and DREVI.

Examples

```
## Not run:
result <- magic(data, t = 3)
dremi <- magic_knnDREMI(result, "GeneA", "GeneB")

## End(Not run)
```

---

magic_optimal_t *Compute Optimal t*

Description

Determines optimal diffusion steps using Procrustes analysis.

Usage

```
magic_optimal_t(
  data,
  diff_op,
  t_max = 20,
  threshold = 0.001,
  subsample_genes = 500,
  plot = FALSE,
  verbose = FALSE
)
```

Arguments

data	Matrix (cells x genes).
diff_op	Diffusion operator.
t_max	Integer. Maximum t to test. Default is 20.
threshold	Numeric. Convergence threshold. Default is 0.001.
subsample_genes	Integer or NULL. Genes to subsample. Default is 500.
plot	Logical. Create diagnostic plot. Default is FALSE.
verbose	Logical or integer. Verbosity level.

Value

A list with t_opt, error_vec, and threshold.

`magic_testdata` *Example Single-Cell Data for MAGICR*

Description

A small example dataset for demonstrating MAGICR functionality. Contains 500 cells and 100 genes with simulated count data.

Usage

```
magic_testdata
```

Format

A matrix with 500 rows (cells) and 100 columns (genes):

rows Cell identifiers (Cell_1 to Cell_500)

columns Gene identifiers (Gene_1 to Gene_100)

values Integer count values (Poisson distributed with lambda=5)

Source

Simulated data for package demonstration

Examples

```
## Not run:
data(magic_testdata)
dim(magic_testdata)
# [1] 500 100

# Run MAGIC on example data
result <- magic(magic_testdata, t = 3)

## End(Not run)
```

magicr_config *Get MAGICR Configuration*

Description

Returns current configuration settings.

Usage

```
magicr_config()
```

Value

Named list of settings

Examples

```
magicr_config()
```

plot *MAGIC Plotting Functions*

Description

Visualization functions for MAGIC results.

plot_magic_genes *Plot Gene-Gene Scatter*

Description

Create a scatter plot of two genes before and after MAGIC.

Usage

```
plot_magic_genes(
  data,
  magic_result,
  gene_x,
  gene_y,
  gene_color = NULL,
  point_size = 1
)
```

Arguments

data	Original data matrix.
magic_result	MAGIC result object.
gene_x	Gene for x-axis.
gene_y	Gene for y-axis.
gene_color	Gene for coloring (optional).
point_size	Point size. Default is 1.

Value

A ggplot object or plots using base R.

Examples

```
## Not run:
data(magic_testdata)
result <- magic(magic_testdata, t = 3)
plot_magic_genes(magic_testdata, result, 1, 2)

## End(Not run)
```

---

preprocessing *Data Preprocessing*

Description

Preprocessing functions for single-cell data.

---

set_magicr_options *Set MAGICR Options*

Description

Configure package-wide options.

Usage

```
set_magicr_options(use_cpp = NULL, verbose_default = NULL)
```

Arguments

use_cpp	Logical. Whether to use C++ acceleration when available.
verbose_default	Integer. Default verbosity level (0, 1, or 2).

Value

Invisibly returns previous settings

Examples

```
## Not run:
# Disable C++ acceleration
set_magicr_options(use_cpp = FALSE)

## End(Not run)
```

`seurat` *Seurat Integration*

Description

Functions for MAGIC integration with Seurat objects (v4 and v5).

`sqrt_normalize` *Square Root Transformation*

Description

Applies square root transformation.

Usage

```
sqrt_normalize(data, verbose = FALSE)
```

Arguments

`data`	Matrix.
`verbose`	Logical. Default is FALSE.

Value

Transformed matrix.

Index