Package 'scPOEM'

August 23, 2025

Type Package

Title Single-Cell Meta-Path Based Omic Embedding

Version 0.1.2

Author Yuntong Hou [aut, cre] (<https://orcid.org/0009-0005-0587-4692>),
Yan Zhong [aut, ctb] (<https://orcid.org/0000-0003-2412-043X>),
Yongjian Yang [ctb] (<https://orcid.org/0000-0002-4135-5014>),
Xinyue Zheng [ctb],
James Cai [ctb] (<https://orcid.org/0000-0002-8081-6725>),
Yeran Chen [ctb],
Youshi Chang [ctb]

Description Provide a workflow to jointly embed chromatin accessibility peaks and expressed genes into a shared low-dimensional space using paired single-cell ATAC-seq (scATAC-seq) and single-cell RNA-seq (scRNA-seq) data. It integrates regulatory relationships among peak-peak interactions (via 'Cicero'), peak-gene interactions (via Lasso, random forest, and XGBoost), and gene-gene interactions (via principal component regression). With the input of paired scATAC-seq and scRNA-seq data matrices, it assigns a low-dimensional feature vector to each gene and peak. Additionally, it supports the reconstruction of gene-gene network with low-dimensional projections (via epsilon-NN) and then the comparison of the networks of two conditions through manifold alignment implemented in 'scTenifoldNet'.

License GPL (>= 2)

Encoding UTF-8

Roxygen list(markdown = TRUE)

Imports
methods, utils, stats, foreach (>= 1.5.2), doParallel (>= 1.0.17), tictoc (>= 1.2.1), Matrix (>= 1.6-3), glmnet (>= 4.1-8), xgboost (>= 1.7.10), reticulate, stringr, magrittr, scTenifoldNet, VGAM (>= 1.1-13), Biobase (>= 2.66.0), BiocGenerics (>= 0.52.0), monocle (>= 2.34.0), cicero (>= 1.24.0)

Depends R (>= 4.1.0)

RoxygenNote 7.3.2

Contents

align_embedding   *Gene Network Reconstruction and Alignment*

Description

Reconstruct gene networks via epsilon-NN and compare conditions using manifold alignment implemented in scTenifoldNet.

Usage

```
align_embedding(
  gene_data1,
  gene_node1,
  E1,
  gene_data2,
  gene_node2,
  E2,
  dirpath = tempdir(),
  save_file = TRUE,
  d = 100
)
```

Arguments

gene_data1	The information for genes in state1, must have a col names "gene_name".
gene_node1	Gene ids that are associated with other peaks or genes in state1.
E1	Embedding representations of peaks and genes in state1.
gene_data2	The information for genes in state2, must have a col names "gene_name".
gene_node2	Gene ids that are associated with other peaks or genes in state2.
E2	Embedding representations of peaks and genes in state2.
dirpath	The folder path to read or write file
save_file	Logical, whether to save the output to a file.
d	The dimension of latent space.

Value

A list containing the following elements:

E_g2 Low-dimensional embedding representations of genes under the two conditions.

common_genes Genes shared between both conditions and used in the analysis.

diffRegulation A list of differential regulatory information for each gene.

Examples

```
library(scPOEM)
library(monocle)
dirpath <- "./example_data"
# Download compare mode example data
data(example_data_compare)
data_S1 <- example_data_compare$S1
data_S2 <- example_data_compare$S2
gg_net1 <- GGN(data_S1$Y, file.path(dirpath, "compare/S1"), save_file=FALSE)
pp_net1 <- PPN(data_S1$X, data_S1$peak_data, data_S1$cell_data,
               data_S1$genome, file.path(dirpath, "compare/S1"), save_file=FALSE)

net_Lasso1 <- PGN_Lasso(data_S1$X, data_S1$Y,
                        data_S1$gene_data, data_S1$neibor_peak,
                        file.path(dirpath, "compare/S1"), save_file=FALSE)
net_RF1 <- PGN_RF(data_S1$X, data_S1$Y, data_S1$gene_data,
                  data_S1$neibor_peak, file.path(dirpath, "compare/S1"), save_file=FALSE)
net_XGB1 <- PGN_XGBoost(data_S1$X, data_S1$Y,
                        data_S1$gene_data, data_S1$neibor_peak,
                        file.path(dirpath, "compare/S1"), save_file=FALSE)
pg_net_list1 <- list(net_Lasso1, net_RF1, net_XGB1)
E_result_S1 <- pg_embedding(gg_net1, pp_net1, pg_net_list1,
                file.path(dirpath, "compare/S1"), save_file=FALSE)

gg_net2 <- GGN(data_S2$Y, file.path(dirpath, "compare/S2"), save_file=FALSE)
pp_net2 <- PPN(data_S2$X, data_S2$peak_data,
               data_S2$cell_data, data_S2$genome,
               file.path(dirpath, "compare/S2"), save_file=FALSE)
net_Lasso2 <- PGN_Lasso(data_S2$X, data_S2$Y,
                        data_S2$gene_data, data_S2$neibor_peak,
                        file.path(dirpath, "compare/S2"), save_file=FALSE)
net_RF2 <- PGN_RF(data_S2$X, data_S2$Y, data_S2$gene_data,
                  data_S2$neibor_peak, file.path(dirpath, "compare/S2"), save_file=FALSE)
net_XGB2 <- PGN_XGBoost(data_S2$X, data_S2$Y,
                        data_S2$gene_data, data_S2$neibor_peak,
                        file.path(dirpath, "compare/S2"), save_file=FALSE)
pg_net_list2 <- list(net_Lasso2, net_RF2, net_XGB2)
E_result_S2 <- pg_embedding(gg_net2, pp_net2, pg_net_list2,
                file.path(dirpath, "compare/S2"), save_file=FALSE)

compare_result <- align_embedding(data_S1$gene_data,
                                  E_result_S1$gene_node,
                                  E_result_S1$E,
                                  data_S2$gene_data,
                                  E_result_S2$gene_node,
                                  E_result_S2$E,
                                  file.path(dirpath, "compare/compare"),
                                  save_file=FALSE)
```

---

eNN   *Network Reconstruction via epsilon-NN*

Description

Reconstruction of gene-gene network via low-dimentional projections (via epsilon-NN).

Usage

```
eNN(E_g)
```

Arguments

E_g   Embedding representations of genes.

Value

The epsilon-NN network.

---

example_data_compare   *Example Input Data for Compare Mode Analysis*

Description

A list containing example single-cell multi-omics data used in "compare" mode of the scPOEM package.

Usage

```
data(example_data_compare)
```

Format

A named list of length 2. Each element is itself a named list with the following components:

X The scATAC-seq data, sparse matrix.

Y The scRNA-seq data, sparse matrix.

peak_data A data.frame containing peak information.

gene_data A data.frame containing gene information (must contain column "gene_name").

cell_data A data.frame containing cell metadata.

neibor_peak The peak IDs within a certain range of each gene, must have cols c("gene_name", "start_use", "end_use"). The id numbers in "start_use" and "end_use" are start from 0.

genome The genome length for the species.

Examples

```
data(example_data_compare)
```

---

example_data_single   *Example Input Data for Single Mode Analysis*

Description

A list containing example single-cell multi-omics data used in "single" mode of the scPOEM package.

Usage

```
data(example_data_single)
```

Format

A named list with 7 elements:

X The scATAC-seq data, sparse matrix.

Y The scRNA-seq data, sparse matrix.

peak_data A data.frame containing peak information.

gene_data A data.frame containing gene information (must contain column "gene_name").

cell_data A data.frame containing cell metadata.

neibor_peak The peak IDs within a certain range of each gene, must have cols c("gene_name", "start_use", "end_use"). The id numbers in "start_use" and "end_use" are start from 0.

genome The genome length for the species.

Examples

```
data(example_data_single)
```

GGN   *Construct Gene-Gene Network*

Description

Construct the gene-gene network via principle component regression.

Usage

```
GGN(
  Y,
  dirpath = tempdir(),
  count_device = 1,
  nComp = 5,
  rebuild_GGN = TRUE,
  save_file = TRUE,
  python_env = "scPOEM_env"
)
```

Arguments

Y	The scRNA-seq data, sparse matrix.
dirpath	The folder path to read or write file.
count_device	The number of cpus used to train the Lasso model.
nComp	The number of PCs used for regression
rebuild_GGN	Logical. Whether to rebuild the gene-gene network (GGN) from scratch. If FALSE, the function will attempt to read from GGN.mtx under dirpath/test in single mode or dirpath/state_name/test in compare mode.
save_file	Logical, whether to save the output to a file.
python_env	Name or path of the Python environment to be used.

Value

The GGN network.

Examples

```
library(scPOEM)
dirpath <- "./example_data"
# Download single mode example data
data(example_data_single)
# Construct GGN net.
gg_net <- GGN(example_data_single$Y,
              file.path(dirpath, "single"),
              save_file=FALSE)
```

PGN_Lasso   *Peak-Gene Network via Lasso*

Description

Construct the peak-gene network via Lasso.

Usage

```
PGN_Lasso(
  X,
  Y,
  gene_data,
  neibor_peak,
  dirpath = tempdir(),
  count_device = 1,
  rebuild_PGN_Lasso = TRUE,
  save_file = TRUE
)
```

Arguments

X	The scATAC-seq data, sparse matrix.
Y	The scRNA-seq data, sparse matrix.
gene_data	The information for genes, must have a col names "gene_name".
neibor_peak	The peak IDs within a certain range of each gene, must have cols c("gene_name", "start_use", "end_use"). The id numbers in "start_use" and "end_use" are start from 0.
dirpath	The folder path to read or write file.
count_device	The number of cpus used to train the Lasso model.
rebuild_PGN_Lasso	
	Logical. Whether to rebuild the peak-gene network via Lasso from scratch. If FALSE, the function will attempt to read from PGN_Lasso.mtx under dirpath/test in single mode or dirpath/state_name/test in compare mode.
save_file	Logical, whether to save the output to a file.

Value

The PGN_Lasso network.

Examples

```
library(scPOEM)
dirpath <- "./example_data"
# Download single mode example data
data(example_data_single)
# Construct PGN net via Lasso.
net_Lasso <- PGN_Lasso(example_data_single$X,
                       example_data_single$Y,
                       example_data_single$gene_data,
                       example_data_single$neibor_peak,
                       file.path(dirpath, "single"),
                       save_file=FALSE)
```

PGN_RF   *Peak-Gene Network via Random Forest*

Description

Construct the peak-gene network via random forest.

Usage

```
PGN_RF(
  X,
  Y,
  gene_data,
  neibor_peak,
  dirpath = tempdir(),
```

```
    count_device = 1,
    rebuild_PGN_RF = TRUE,
    save_file = TRUE,
    seed = NULL,
    python_env = "scPOEM_env"
)
```

Arguments

X	The scATAC-seq data, sparse matrix.
Y	The scRNA-seq data, sparse matrix.
gene_data	The information for genes, must have a col names "gene_name".
neibor_peak	The peak IDs within a certain range of each gene, must have cols c("gene_name", "start_use", "end_use"). The id numbers in "start_use" and "end_use" are start from 0.
dirpath	The folder path to read or write file.
count_device	The number of cpus used to train the Lasso model.
rebuild_PGN_RF	Logical. Whether to rebuild the peak-gene network via random forest from scratch. If FALSE, the function will attempt to read from `PGN_RF.mtx` under `dirpath/test` in single mode or `dirpath/state_name/test` in compare mode.
save_file	Logical, whether to save the output to a file.
seed	An integer specifying the random seed to ensure reproducible results.
python_env	Name or path of the Python environment to be used.

Value

The PGN_RF network.

Examples

```
library(scPOEM)
dirpath <- "./example_data"
# Download single mode example data
data(example_data_single)
# Construct PGN net via random forest (RF).
net_RF <- PGN_RF(example_data_single$X,
                 example_data_single$Y,
                 example_data_single$gene_data,
                 example_data_single$neibor_peak,
                 file.path(dirpath, "single"),
                 save_file=FALSE)
```

PGN_XGBoost *Peak-Gene Network via XGBoost*

Description

Construct the peak-gene network via XGBoost.

Usage

```
PGN_XGBoost(
  X,
  Y,
  gene_data,
  neibor_peak,
  dirpath = tempdir(),
  count_device = 1,
  rebuild_PGN_XGB = TRUE,
  save_file = TRUE
)
```

Arguments

X	The scATAC-seq data, sparse matrix.
Y	The scRNA-seq data, sparse matrix.
gene_data	The information for genes, must have a col names "gene_name".
neibor_peak	The peak IDs within a certain range of each gene, must have cols c("gene_name", "start_use", "end_use"). The id numbers in "start_use" and "end_use" are start from 0.
dirpath	The folder path to read or write file.
count_device	The number of cpus used to train the Lasso model.
rebuild_PGN_XGB	
	Logical. Whether to rebuild the peak-gene network via XGBoost from scratch. If FALSE, the function will attempt to read from PGN_XGB.mtx under dirpath/test in single mode or dirpath/state_name/test in compare mode.
save_file	Logical, whether to save the output to a file.

Value

The PGN_XGBoost network.

Examples

```
library(scPOEM)
dirpath <- "./example_data"
# Download single mode example data
data(example_data_single)
# Construct PGN net via XGBoost.
net_XGB <- PGN_XGBoost(example_data_single$X,
                       example_data_single$Y,
                       example_data_single$gene_data,
```

```
                    example_data_single$neibor_peak,
                    file.path(dirpath, "single"),
                    save_file=FALSE)
```

---

pg_embedding   *Co-embeddings of Peaks and Genes.*

Description

Learn the low-dimensional representations for peaks and genes with a meta-path based method.

Usage

```
pg_embedding(
  gg_net,
  pp_net,
  pg_net_list,
  dirpath = tempdir(),
  relearn_pg_embedding = TRUE,
  save_file = TRUE,
  d = 100,
  numwalks = 5,
  walklength = 3,
  epochs = 100,
  neg_sample = 5,
  batch_size = 32,
  weighted = TRUE,
  exclude_pos = FALSE,
  seed = NULL,
  python_env = "scPOEM_env"
)
```

Arguments

gg_net	The gene-gene network.
pp_net	The peak-peak network.
pg_net_list	A list of peak-gene networks, constructed via different methods.
dirpath	The folder path to read or write file.
relearn_pg_embedding	
	Logical. Whether to relearn the low-dimensional representations for peaks and genes from scratch. If FALSE, the function will attempt to read from node_embeddings.mtx, node_used_peak.csv, node_used_gene.csv under dirpath/embedding in single mode or dirpath/state_name/embedding in compare mode.
save_file	Logical, whether to save the output to a file.
d	Dimension of the latent space. Default is 100.
numwalks	Number of random walks per node. Default is 5.

walklength	Length of walk depth. Default is 3.
epochs	Number of training epochs. Default is 100.
neg_sample	Number of negative samples per positive sample. Default is 5.
batch_size	Batch size for training. Default is 32.
weighted	Whether the sampling network is weighted. Default is TRUE.
exclude_pos	Whether to exclude positive samples from negative sampling. Default is FALSE.
seed	An integer specifying the random seed to ensure reproducible results.
python_env	Name or path of the Python environment to be used.

Value

A list containing the following:

E Low-dimensional representations of peaks and genes

peak_node Peak ids that are associated with other peaks or genes.

gene_node Gene ids that are associated with other peaks or genes.

Examples

```
library(scPOEM)
library(monocle)
dirpath <- "./example_data"
# Download single mode example data
data(example_data_single)
gg_net <- GGN(example_data_single$Y,
              file.path(dirpath, "single"),
              save_file=FALSE)
pp_net <- PPN(example_data_single$X, example_data_single$peak_data,
              example_data_single$cell_data, example_data_single$genome,
              file.path(dirpath, "single"), save_file=FALSE)
net_Lasso <- PGN_Lasso(example_data_single$X, example_data_single$Y,
                       example_data_single$gene_data, example_data_single$neibor_peak,
                       file.path(dirpath, "single"), save_file=FALSE)
net_RF <- PGN_RF(example_data_single$X, example_data_single$Y,
                 example_data_single$gene_data, example_data_single$neibor_peak,
                 file.path(dirpath, "single"), save_file=FALSE)
net_XGB <- PGN_XGBoost(example_data_single$X, example_data_single$Y,
                       example_data_single$gene_data, example_data_single$neibor_peak,
                       file.path(dirpath, "single"), save_file=FALSE)
E_result <- pg_embedding(gg_net, pp_net, list(net_Lasso, net_RF, net_XGB),
                         file.path(dirpath, "single"), save_file=FALSE)
```

PPN   *Construct Peak-Peak Network*

Description

Construct peak-peak network.

Usage

```
PPN(
  X,
  peak_data,
  cell_data,
  genome,
  dirpath = tempdir(),
  rebuild_PPN = TRUE,
  save_file = TRUE,
  seed = NULL
)
```

Arguments

X	The scATAC-seq data, sparse matrix.
peak_data	The information for peaks, must have a col names "peak_name".
cell_data	The information for cells, must have a col names "cell_name".
genome	The genome length for the species.
dirpath	The folder path to read or write file.
rebuild_PPN	Logical. Whether to rebuild the peak-peak network (PPN) from scratch. If FALSE, the function will attempt to read from `PPN.mtx` under `dirpath/test` in `single` mode or `dirpath/state_name/test` in `compare` mode.
save_file	Logical, whether to save the output to a file.
seed	An integer specifying the random seed to ensure reproducible results.

Value

The PPN network.

Examples

```
library(scPOEM)
library(monocle)
dirpath <- "./example_data"
# Download single mode example data
data(example_data_single)
# Construct PPN net.
pp_net <- PPN(example_data_single$X,
              example_data_single$peak_data,
              example_data_single$cell_data,
              example_data_single$genome,
              file.path(dirpath, "single"),
              save_file=FALSE)
```

---

scPOEM   *Main Function.*

Description

This function takes paired single-cell ATAC-seq (scATAC-seq) and RNA-seq (scRNA-seq) data
to embed peaks and genes into a shared low-dimensional space. It integrates regulatory relation-
ships from peak-peak interactions (via Cicero), peak-gene interactions (via Lasso, random forest,
and XGBoost), and gene-gene interactions (via principal component regression). Additionally, it
supports gene-gene network reconstruction using epsilon-NN projections and compares networks
across conditions through manifold alignment (scTenifoldNet).

Usage

```
scPOEM(
  mode = c("single", "compare"),
  input_data,
  dirpath = tempdir(),
  count_device = 1,
  nComp = 5,
  seed = NULL,
  numwalks = 5,
  walklength = 3,
  epochs = 100,
  neg_sample = 5,
  batch_size = 32,
  weighted = TRUE,
  exclude_pos = FALSE,
  d = 100,
  rebuild_GGN = TRUE,
  rebuild_PPN = TRUE,
  rebuild_PGN_Lasso = TRUE,
  rebuild_PGN_RF = TRUE,
  rebuild_PGN_XGB = TRUE,
  relearn_pg_embedding = TRUE,
  save_file = TRUE,
  pg_method = c("Lasso", "RF", "XGBoost"),
  python_env = "scPOEM_env"
)
```

Arguments

mode
: The mode indicating whether to analyze data from a single condition or to com-
pare two conditions.

input_data
: A list of input data.

  If mode = "single", input_data must be a list containing the following **seven
  objects**:

  - X: The scATAC-seq data, sparse matrix.
  - Y: The scRNA-seq data, sparse matrix.
  - peak_data: A data.frame containing peak information.

- gene_data: A data.frame containing gene information (must contain a column "gene_name").
- cell_data: A data.frame containing cell metadata.
- neibor_peak: The peak IDs within a certain range of each gene, must have cols c("gene_name", "start_use", "end_use"). The id numbers in "start_use" and "end_use" are start from 0.
- genome: The genome length for the species.

If mode = "compare", input_data must be a **named list of two elements**, with names corresponding to two state names (e.g., "S1" and "S2"). Each element must itself be a list containing the same seven components as described above for mode = "single".

dirpath	The folder path to read or write file.
count_device	The number of cpus used to train models.
nComp	The number of PCs used for regression in constructing GGN.
seed	An integer specifying the random seed to ensure reproducible results.
numwalks	Number of random walks per node. Default is 5.
walklength	Length of walk depth. Default is 3.
epochs	Number of training epochs. Default is 100.
neg_sample	Number of negative samples per positive sample. Default is 5.
batch_size	Batch size for training. Default is 32.
weighted	Whether the sampling network is weighted. Default is TRUE.
exclude_pos	Whether to exclude positive samples from negative sampling. Default is FALSE.
d	The dimension of latent space. Default is 100.
rebuild_GGN	Logical. Whether to rebuild the gene-gene network from scratch. If FALSE, the function will attempt to read from GGN.mtx under dirpath/test in single mode or dirpath/state_name/test in compare mode.
rebuild_PPN	Logical. Whether to rebuild the peak-peak network from scratch. If FALSE, the function will attempt to read from PPN.mtx under dirpath/test in single mode or dirpath/state_name/test in compare mode.
rebuild_PGN_Lasso	
	Logical. Whether to rebuild the peak-gene network via Lasso from scratch. If FALSE, the function will attempt to read from PGN_Lasso.mtx under dirpath/test in single mode or dirpath/state_name/test in compare mode.
rebuild_PGN_RF	Logical. Whether to rebuild the peak-gene network via random forest from scratch. If FALSE, the function will attempt to read from PGN_RF.mtx under dirpath/test in single mode or dirpath/state_name/test in compare mode.
rebuild_PGN_XGB	
	Logical. Whether to rebuild the peak-gene network via XGBoost from scratch. If FALSE, the function will attempt to read from PGN_XGB.mtx under dirpath/test in single mode or dirpath/state_name/test in compare mode.
relearn_pg_embedding	
	Logical. Whether to relearn the low-dimensional representations for peaks and genes from scratch. If FALSE, the function will attempt to read from node_embeddings.mtx, node_used_peak.csv, node_used_gene.csv under dirpath/embedding in single mode or dirpath/state_name/embedding in compare mode.

save_file	Logical, whether to save the output to a file.
pg_method	The vector of methods used to construct peak-gene net. Default is c("Lasso", "RF", "XGBoost").
python_env	Name or path of the Python environment to be used.

Value

The scPOEM result.

Single Mode Returns a list containing the following elements:

    E Low-dimensional representations of peaks and genes.

    peak_node Peak IDs that are associated with other peaks or genes.

    gene_node Gene IDs that are associated with other peaks or genes.

Compare Mode Returns a list containing the following elements:

    state1 name The single-mode result for the first condition.

    state2 name The single-mode result for the second condition.

    compare A summary list containing:

        E_g2 Low-dimensional embedding representations of genes under the two conditions.

        common_genes Genes shared between both conditions and used in the analysis.

        diffRegulation A list of differential regulatory information for each gene.

Examples

```
library(scPOEM)
library(monocle)
dirpath <- "./example_data"
# An example for analysing a single dataset.
# Download and read data.
data(example_data_single)
single_result <- scPOEM(mode = "single",
                        input_data=example_data_single,
                        dirpath=file.path(dirpath, "single"),
                        save_file=FALSE)

# An example for analysing and comparing datasets from two conditions.
# Download compare mode example data
data(example_data_compare)
compare_result <- scPOEM(mode = "compare",
                         input_data=example_data_compare,
                         dirpath=file.path(dirpath, "compare"),
                         save_file=FALSE)
```

Index