Package 'ExpressionCellNet'

March 4, 2026

Type Package

Title Network-Based Analysis of Gene Expression Perturbations

Version 0.1.1

Description Network-centric framework for integrative analysis of high-throughput gene expression data using user-supplied gene-gene interaction graphs. Constructs seed-centered multi-generation networks constrained by expression correlations and simulates expression perturbation scenarios via regression-based prediction (van Dam, 2018).

URL https://doi.org/10.1093/bib/bbw139

License GPL-3

Encoding UTF-8

LazyData true

Depends R (>= 4.1.0)

Imports igraph, ggplot2, uwot, Metrics, crayon

RoxygenNote 7.3.3

NeedsCompilation no

Author Ehsan Keramati [aut, cre]

Maintainer Ehsan Keramati <ehsan.keramati14@gmail.com>

Repository CRAN

Date/Publication 2026-03-04 09:50:13 UTC

Contents

`AnalyzeNetwork` *Analyze network topology and identify hub genes*

Description

Computes degree, closeness, betweenness, and eigenvector centrality and stores: (i) Network Analysis table and (ii) Hub Genes table (top 10

Usage

```
AnalyzeNetwork(ExpCellNetObj)
```

Arguments

`ExpCellNetObj`	An ExpressionCellNet object.

Value

Updated ExpressionCellNet object with "Network Analysis" and "Hub Genes".

Examples

```
data("example_data")
obj <- AnalyzeNetwork(example_data$expcellnetobj)
```

`BuildNetwork` *Construct a gene interaction network centered on a seed gene*

Description

Builds a multi-generation network starting from a seed gene using curated interactions and correlation filtering across generations.

Usage

```
BuildNetwork(
  ExpCellNetObj,
  Gene = "Gene1",
  Generation = 3,
  CorrelationThreshold = c(0.7, 0.7, 0.7)
)
```

Arguments

`ExpCellNetObj`	An ExpressionCellNet object containing expression and interaction data.
`Gene`	Character; seed gene name (generation 0).
`Generation`	Integer; number of generations to expand from the seed gene.
`CorrelationThreshold`	Numeric vector; per-generation correlation thresholds.

Value

Updated ExpressionCellNet object with added components such as "Network" and "Generations".

Examples

```
data("example_data")
obj <- BuildNetwork(
    example_data$expcellnetobj,
    Gene = "Gene1",
    Generation = 3,
    CorrelationThreshold = c(0.65, 0.70, 0.75)
)
```

CountMatrixNormalization *Normalize expression count matrix*

Description

Filters low-expression genes (optional), applies CPM normalization (optional), and log2 transformation (optional). Stores results inside the ExpressionCellNet object.

Usage

```
CountMatrixNormalization(
  ExpCellNetObj,
  CheckExpression = TRUE,
  CPM = TRUE,
  LogTransform = FALSE
)
```

Arguments

ExpCellNetObj	An ExpressionCellNet object.
CheckExpression	Logical; if TRUE, low-expression genes are filtered.
CPM	Logical; if TRUE, counts are converted to counts-per-million.
LogTransform	Logical; if TRUE, log2 transform is applied (typically after CPM).

Value

Updated ExpressionCellNet object containing normalized matrix.

Examples

```
data("example_data")
obj <- CountMatrixNormalization(
    example_data$expcellnetobj,
    CheckExpression = TRUE,
    CPM = TRUE,
    LogTransform = TRUE
)
```

createExpCellNetObj *Create an ExpressionCellNet object*

Description

Wraps the core inputs (count matrix, annotation table, and interaction database) into a single ExpressionCellNet object used by downstream functions.

Usage

```
createExpCellNetObj(count_matrix, annotations, intraction_db)
```

Arguments

count_matrix	A gene-by-sample expression count matrix (rows: genes, cols: samples).
annotations	A data.frame containing gene annotations/mapping information.
intraction_db	A data.frame of gene-gene interactions (two columns).

Value

An ExpressionCellNet object (list) containing core data components.

Examples

```
data("example_data")
obj <- createExpCellNetObj(
   count_matrix  = example_data$counts,
   annotations   = example_data$annot_df,
   intraction_db = example_data$interactions
)
```

example_data *Example dataset for ExpressionCellNet*

Description

Simulated example dataset containing four objects: `counts`, `annot_df`, `interactions`, and `expcellnetobj`. All data are mock / artificial for testing.

Usage

```
data(example_data)
```

Format

An RData file containing:

counts Numeric matrix of simulated RNA-seq counts.

annot_df Data frame with ENSEMBL IDs and Gene Symbols.

interactions Data frame representing gene–gene interactions.

expcellnetobj Example ExpressionCellNet object.

`ExtractFeature` *Extract network-derived expression features*

Description

Extracts a feature matrix from the normalized expression data by restricting genes to those present in the constructed interaction network. The resulting matrix is intended for downstream modeling and dimensionality reduction.

Usage

```
ExtractFeature(ExpCellNetObj)
```

Arguments

`ExpCellNetObj`	An ExpressionCellNet object containing a normalized count matrix and a constructed network.

Value

A data.frame or matrix of network-derived expression features (genes × samples), suitable for regression or multivariate analysis.

Examples

```
data("example_data")
features <- ExtractFeature(example_data$expcellnetobj)
```

FindPathway *Find the shortest path between two genes in the network*

Description

Computes the shortest interaction path (as a sequence of gene names) between a start gene and an end gene using the network stored in the ExpressionCellNet object.

Usage

```
FindPathway(ExpCellNetObj, Start_gene, End_gene)
```

Arguments

ExpCellNetObj	An ExpressionCellNet object containing a constructed network.
Start_gene	Character; source gene symbol (must exist as a node in the network).
End_gene	Character; destination gene symbol (must exist as a node in the network).

Value

A character vector of gene symbols representing the shortest path.

Examples

```
data("example_data")
path <- FindPathway(example_data$expcellnetobj, Start_gene="Gene1", End_gene="Gene11")
path
```

MeanExpression *Compute mean expression of a gene*

Description

Computes mean expression from raw or normalized matrix (default: normalized).

Usage

```
MeanExpression(ExpCellNetObj, Gene, Normalized = TRUE)
```

Arguments

ExpCellNetObj	An ExpressionCellNet object.
Gene	Character; target gene.
Normalized	Logical; use normalized matrix (TRUE) or raw (FALSE).

Value

Numeric mean expression value.

Examples

```
data("example_data")
MeanExpression(example_data$expcellnetobj , "Gene1")
```

`NetworkPrediction` *Predict network-wide expression changes after perturbing a seed gene*

Description

Fits linear regression models for seed-to-network genes and predicts shifts under a user-defined perturbation scenario. Stores multiple result components in the object.

Usage

```
NetworkPrediction(ExpCellNetObj, Gene, Expression)
```

Arguments

`ExpCellNetObj`	An ExpressionCellNet object.
`Gene`	Character; seed gene to perturb.
`Expression`	Numeric; imposed expression value under perturbation.

Value

Updated ExpressionCellNet object with regression models and results.

Examples

```
data("example_data")
obj <- NetworkPrediction(example_data$expcellnetobj, Gene="Gene1", Expression=0)
```

PCAforGenes *PCA on network-derived features*

Description

Performs principal component analysis (PCA) on the network-restricted feature matrix returned by ExtractFeature(), after min-max scaling, and returns a ggplot of PC1 vs PC2.

Usage

```
PCAforGenes(ExpCellNetObj)
```

Arguments

ExpCellNetObj	An ExpressionCellNet object containing "Network", "Generations", and a normalized count matrix.

Value

A ggplot object showing PCA embedding of genes (nodes).

Examples

```
data("example_data")
p <- PCAforGenes(example_data$expcellnetobj)
p
```

PlotCorrelation *Plot correlation between two genes*

Description

Plots correlation/regression between two genes (e.g., seed and distal gene).

Usage

```
PlotCorrelation(ExpCellNetObj, Gene1, Gene2)
```

Arguments

ExpCellNetObj	An ExpressionCellNet object.
Gene1	Character; first gene.
Gene2	Character; second gene.

Value

Invisibly returns NULL (plot is produced).

Examples

```
data("example_data")
PlotCorrelation(example_data$expcellnetobj, "Gene1", "Gene11")
```

`PlotCountMatrix` *Plot expression matrix distribution*

Description

Visualizes the distribution of raw or normalized expression values across samples.

Usage

```
PlotCountMatrix(ExpCellNetObj, Matrix = "Normalized")
```

Arguments

`ExpCellNetObj`	An ExpressionCellNet object.
`Matrix`	Character; which matrix to plot (e.g., "RAW" or "Normalized").

Value

A plot (base R) is produced as a side effect.

Examples

```
data("example_data")
PlotCountMatrix(example_data$expcellnetobj, Matrix = "RAW")
PlotCountMatrix(example_data$expcellnetobj, Matrix = "Normalized")
```

PlotNetworkPrediction *Plot network prediction summary*

Description

Creates a summary plot of predicted percent changes across network genes (bar plot).

Usage

```
PlotNetworkPrediction(ExpCellNetObj)
```

Arguments

ExpCellNetObj	An ExpressionCellNet object after NetworkPrediction().

Value

Invisibly returns NULL (plot is produced).

Examples

```
data("example_data")
PlotNetworkPrediction(example_data$expcellnetobj)
```

PlotPathwayPrediction *Plot pathway-restricted prediction to a target gene*

Description

Summarizes modeled expression changes along the seed-to-target pathway.

Usage

```
PlotPathwayPrediction(ExpCellNetObj, target_gene)
```

Arguments

ExpCellNetObj	An ExpressionCellNet object after NetworkPrediction().
target_gene	Character; destination gene to focus on.

Value

Invisibly returns NULL (plot is produced).

Examples

```
data("example_data")
obj <- NetworkPrediction(example_data$expcellnetobj, Gene="Gene1", Expression=0)
PlotPathwayPrediction(obj, "Gene11")
```

`RegenerateTrainTest` *Regenerate train/test split*

Description

Re-creates the internal random 80/20 train-test split without stratifimessageion.

Usage

```
RegenerateTrainTest(ExpCellNetObj)
```

Arguments

`ExpCellNetObj`	An ExpressionCellNet object.

Value

Updated ExpressionCellNet object with regenerated split.

Examples

```
data("example_data")
obj <- RegenerateTrainTest(example_data$expcellnetobj)
```

`ShowHubGenes` *Show hub genes on the network*

Description

Highlights hub genes on the igraph network visualization.

Usage

```
ShowHubGenes(ExpCellNetObj)
```

Arguments

`ExpCellNetObj`	An ExpressionCellNet object after AnalyzeNetwork().

Value

Invisibly returns NULL (plot is produced).

Examples

```
data("example_data")
ShowHubGenes(example_data$expcellnetobj)
```

ShowNetwork *Visualize the constructed network*

Description

Plots the current network stored in the ExpressionCellNet object.

Usage

```
ShowNetwork(ExpCellNetObj)
```

Arguments

ExpCellNetObj	An ExpressionCellNet object containing a "Network" edge table.

Value

A plot (igraph/base) is produced as a side effect.

Examples

```
data("example_data")
ShowNetwork(example_data$expcellnetobj)
```

ShowPathway *Visualize a pathway (highlighted subgraph)*

Description

Highlights the nodes/edges of a provided pathway on the full network visualization.

Usage

```
ShowPathway(ExpCellNetObj, Pathway)
```

Arguments

ExpCellNetObj	An ExpressionCellNet object containing a constructed network.
Pathway	A character vector of genes representing a path (e.g., output of FindPathway()).

Value

A plot (igraph/base) is produced as a side effect.

Examples

```
data("example_data")
p <- FindPathway(example_data$expcellnetobj, Start_gene="Gene1", End_gene="Gene11")
ShowPathway(example_data$expcellnetobj, Pathway = p)
```

UMAPforGenes *UMAP on network-derived features*

Description

Computes a 2D UMAP embedding of the network-restricted feature matrix returned by ExtractFeature() (after min-max scaling) and returns a ggplot visualization.

Usage

```
UMAPforGenes(ExpCellNetObj)
```

Arguments

ExpCellNetObj	An ExpressionCellNet object containing "Network", "Generations", and a normalized count matrix.

Value

A ggplot object showing UMAP embedding of genes (nodes).

Examples

```
data("example_data")
u <- UMAPforGenes(example_data$expcellnetobj)
u
```

Index