Package: GraphExperiment (via r-universe)

May 29, 2026

Title S4 Class for Quantitative Data and Associated Networks

Version 1.0.1

Date 2026-03-01

Description GraphExperiment provides users and developers with an S4 class that extends `SingleCellExperiment` by offering infrastructure to store and retrieve networks (`igraph` objects) representing how assay features and/or observations are associated with each other. The class was designed to store networks inferred from high-dimensional quantitative data, with feature-feature networks including gene coexpression networks (GCNs), gene regulatory networks (GRNs), and co-abundance networks (from proteomics and metabolomics), and observation-observation network including cell-cell distances, species-species relationships, and sample-sample similarities.

License GPL-3

URL https://github.com/almeidasilvaf/GraphExperiment

BugReports https://support.bioconductor.org/tag/GraphExperiment

Encoding UTF-8

LazyData false

biocViews DataRepresentation, DataImport, Infrastructure, GeneExpression, Transcriptomics, Network, SingleCell

Depends SingleCellExperiment, igraph

Imports methods, SummarizedExperiment, BiocBaseUtils, S4Vectors

Suggests knitr, BiocStyle, testthat, rmarkdown, covr, sessioninfo

Config/testthat/edition 3

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

VignetteBuilder knitr

Config/pak/sysreqs libglpk-dev libxml2-dev zlib1g-dev

Repository https://bioc-release.r-universe.dev

Date/Publication 2026-05-22 19:37:38 UTC

RemoteUrl https://github.com/bioc/GraphExperiment

RemoteRef RELEASE_3_23

RemoteSha 4494c104f4aaf54a5f06c4d872a566b86c34049b

Contents

`GraphExperiment-class` `GraphExperiment` *S4 class*

Description

The `GraphExperiment` class was designed to represent rectangular, quantitative data (e.g., from transcriptomics, proteomics, metabolomics) along with graphs showing how features (e.g., genes, proteins, compounds) and observations (e.g., samples, cells, species, spots) interact with each other. It extends `SingleCellExperiment` by providing users with additional slots where row/column graphs can be stored.

Usage

```
GraphExperiment(..., rowGraphs = list(), colGraphs = list())
```

Arguments

`...`	Arguments passed to the `SingleCellExperiment` constructor function.
`rowGraphs`	A list of `igraph` objects with one or multiple graphs representing how features relate to each other. Node names (i.e., `V(graph)$name`) must match rownames of assays.
`colGraphs`	A list of `igraph` objects with one or multiple graphs representing how observations relate to each other. Node names (i.e., `V(graph)$name`) must match colnames of assays.

Details

Like `SingleCellExperiment`, the `GraphExperiment` S4 class stores quantitative data with associated metadata (i.e., `rowData` and `colData`) along with embeddings from dimensionality reduction techniques. However, it provides users with additional containers for `igraph` objects containing graphs describing how features and/or observations interact with each other. Graphs for features are stored in a `rowGraphs` slot, and graphs for observations are stored in a `colGraphs` slot. Both slots

can hold one or multiple `igraph` objects with some sort of network representation of the features in `rownames` or observations in `colnames`. Example graphs for features can be coexpression networks, regulatory networks, or co-abundance networks. Example graphs for observations can be cell-cell distances, sample-sample distances, or species networks representing genealogies (as an alternative to phylogenies).

Importantly, node names in each row graph must match `rownames` of the assays, and node names in each column graph must match `colnames` of the assays. Besides, subsetting methods simultaneously subset `assays`, `rowData`, `rowGraphs`, `colData`, and `colGraphs`.

Besides the constructor function (`GraphExperiment()`), a `GraphExperiment` object can also be created by coercing from a `SummarizedExperiment` or `SingleCellExperiment` object.

Value

A `GraphExperiment` object.

Examples

```
# Example 1: from constructor function ----
## Simulate a matrix with 200 genes and 100 cells
gene_ids <- paste0("gene", seq_len(200))
cell_ids <- paste0("cell", seq_len(100))
mat <- matrix(rpois(20000, 5), ncol = 100, dimnames = list(gene_ids, cell_ids))

## Create a rowGraph from correlations (`igraph` object)
g <- graph_from_adjacency_matrix(cor(t(mat)), weighted = TRUE)

## Create a colGraph, also from correlations (but see scran::buildSNNGraph)
g2 <- graph_from_adjacency_matrix(cor(mat), weighted = TRUE)

## Construct `GraphExperiment` object
ge <- GraphExperiment(
    assays = list(counts = mat),
    rowGraphs = list(cor = g),
    colGraphs = list(cellcor = g2)
)
ge

# Example 2: From `SingleCellExperiment` object ----
sce <- SingleCellExperiment(assays = list(counts = mat))
ge <- as(sce, "GraphExperiment")
ge
```

`GraphExperiment-coerce`

Coercion methods for `GraphExperiment` *objects*

Description

The GraphExperiment class inherits from SingleCellExperiment, which in turn inherits from (Ranged)SummarizedExperiment. To ensure GraphExperiment easily interoperates with these classes, we provide users with traditional as() coercion methods.

Arguments

from	An object of class SingleCellExperiment, SummarizedExperiment, or RangedSummarizedExperiment to be coerced to GraphExperiment.

Value

A GraphExperiment object.

Examples

```
# Simulate a count matrix
gene_ids <- paste0("gene", seq_len(200))
cell_ids <- paste0("cell", seq_len(100))
mat <- matrix(rpois(20000, 5), ncol = 100, dimnames = list(gene_ids, cell_ids))

# Coerce from `SummarizedExperiment`
se <- SummarizedExperiment(assays = list(mat))
as(se, "GraphExperiment")

# Coerce from `RangedSummarizedExperiment`
rse <- as(se, "RangedSummarizedExperiment")
as(rse, "GraphExperiment")

# Coerce from `SingleCellExperiment`
sce <- as(se, "SingleCellExperiment")
as(sce, "GraphExperiment")
```

GraphExperiment-methods

Methods for GraphExperiment *objects*

Description

The GraphExperiment class provides users with methods to get and set graphs (igraph objects) representing how features and observations of SingleCellExperiment objects relate to each other.

Usage

```
## S4 method for signature 'GraphExperiment'
rowData(x, use.names = TRUE, ...)

## S4 method for signature 'GraphExperiment'
```

```
colData(x, use.names = TRUE, ...)

## S4 method for signature 'GraphExperiment'
rowGraphs(x)

## S4 method for signature 'GraphExperiment'
colGraphs(x)

## S4 method for signature 'GraphExperiment,missing'
rowGraph(x, i)

## S4 method for signature 'GraphExperiment,ANY'
rowGraph(x, i)

## S4 method for signature 'GraphExperiment,missing'
colGraph(x, i)

## S4 method for signature 'GraphExperiment,ANY'
colGraph(x, i)

## S4 method for signature 'GraphExperiment'
rowGraphNames(x)

## S4 method for signature 'GraphExperiment'
colGraphNames(x)

## S4 replacement method for signature 'GraphExperiment'
rowGraphs(x) <- value

## S4 replacement method for signature 'GraphExperiment'
colGraphs(x) <- value

## S4 replacement method for signature 'GraphExperiment'
rowGraph(x, i) <- value

## S4 replacement method for signature 'GraphExperiment'
colGraph(x, i) <- value

## S4 replacement method for signature 'GraphExperiment,character'
rowGraphNames(x) <- value

## S4 replacement method for signature 'GraphExperiment,character'
colGraphNames(x) <- value
```

Arguments

x	A GraphExperiment object.
use.names	Passed to the colData method of SingleCellExperiment. Default: TRUE.

...	Ignored.
i	List element (numeric for index, character for name) of the element to access or replace.
value	Replacement value for replacement methods.

Value

Return values depend on the method. See details and examples.

rowGraphs/colGraphs and rowGraph/colGraph methods

`rowGraphs(x)` **and** `colGraphs(x)`**:** Getter for a `SimpleList` of `igraph` objects representing rows and columns, respectively.

`rowGraphs(x) <- value` **and** `colGraphs(x) <- value`**:** Setter for a SimpleList or list (coerced to SimpleList) of `igraph` objects representing rows and columns, respectively.

`rowGraph(x, i)` **and** `colGraph(x, i)`**:** Getter for an `igraph` object containing graph `i` from the list stored in `rowGraphs` and `colGraphs`, respectively.

`rowGraph(x, i) <- value` **and** `colGraph(x, i) <- value`**:** Setter for an `igraph` object to be stored in element `i` of `rowGraphs` and `colGraphs`.

rowGraphNames and colGraphNames methods

`rowGraphNames(x)` **and** `colGraphNames(x)`**:** Getter to extract names of graphs in `rowGraphs` and `colGraphs`.

`rowGraphNames(x) <- value` **and** `colGraphNames(x) <- value`**:** Setter to assign new names to the graphs stored in `rowGraphs` and `colGraphs`.

rowData method

`rowData(x)`**:** Getter to extract rowData (as in `SingleCellExperiment` objects), but with node attributes of graphs included.

colData method

`colData(x)`**:** Getter to extract colData (as in `SingleCellExperiment` objects), but with node attributes of graphs included.

Examples

```
# Simulate elements of a GraphExperiment object
## Assays
gene_ids <- paste0("gene", seq_len(200))
cell_ids <- paste0("cell", seq_len(100))
mat <- matrix(rpois(20000, 5), ncol = 100, dimnames = list(gene_ids, cell_ids))

## rowGraph (with node attributes)
g <- graph_from_adjacency_matrix(cor(t(mat)), weighted = TRUE)
V(g)$degree <- igraph::strength(g)
```

```
## colGraph
g2 <- graph_from_adjacency_matrix(cor(mat), weighted = TRUE)

## rowData
rdata <- data.frame(
    row.names = gene_ids,
    pathway = sample(c("P1", "P2"), size = length(gene_ids), replace = TRUE),
    coding = sample(c(TRUE, FALSE), size = length(gene_ids), replace = TRUE)
)

## colData
cdata <- data.frame(
    row.names = cell_ids,
    celltype = sample(c("ct1", "ct2"), size = length(cell_ids), replace = TRUE)
)


# Create a GraphExperiment object
ge <- GraphExperiment(
    assays = list(counts = mat),
    rowData = rdata,
    colData = cdata,
    rowGraphs = list(cor = g),
    colGraphs = list(cellcor = g2)
)
ge

# Extract graph names
rowGraphNames(ge)
colGraphNames(ge)


# Extract graphs
rowGraphs(ge)
rowGraph(ge, "cor")

colGraphs(ge)
colGraph(ge, "cellcor")


# Add a new graph
rowGraph(ge, "newcor") <- g
ge

# Add a list of graphs
colGraphs(ge) <- list(cellcor = g2, new_cellcor = g2)
ge

# Replace graph names
rowGraphNames(ge) <- c("network1", "network2")
ge

# Access rowData (note: rowData + node attributes combined)
```

```
rowData(ge)

# Access colData (note: colData + node attributes combined)
colData(ge)
```

GraphExperiment-subset

Subsetting GraphExperiment *objects*

Description

The subsetting method for GraphExperiment objects ensures that nodes from igraph objects are filtered to match rows and columns with the remainder of the object.

Arguments

x	A GraphExperiment object.
i	Numeric, row indices for subsetting.
j	Numeric, column indices for subsetting.

Value

a GraphExperiment object.

subset

[: subsetting method

Examples

```
# Simulate elements of a GraphExperiment object
## Assays
gene_ids <- paste0("gene", seq_len(200))
cell_ids <- paste0("cell", seq_len(100))
mat <- matrix(rpois(20000, 5), ncol = 100, dimnames = list(gene_ids, cell_ids))

## rowGraph (with node attributes)
g <- graph_from_adjacency_matrix(cor(t(mat)), weighted = TRUE)
V(g)$degree <- igraph::strength(g)

## colGraph
g2 <- graph_from_adjacency_matrix(cor(mat), weighted = TRUE)

## rowData
rdata <- data.frame(
    row.names = gene_ids,
    pathway = sample(c("P1", "P2"), size = length(gene_ids), replace = TRUE),
    coding = sample(c(TRUE, FALSE), size = length(gene_ids), replace = TRUE)
```

```
)

## colData
cdata <- data.frame(
    row.names = cell_ids,
    celltype = sample(c("ct1", "ct2"), size = length(cell_ids), replace = TRUE)
)


# Create a GraphExperiment object
ge <- GraphExperiment(
    assays = list(counts = mat),
    rowData = rdata,
    colData = cdata,
    rowGraphs = list(cor = g),
    colGraphs = list(cellcor = g2)
)
ge

# Subset object
ge[1:5, 1:5]
```

Index