Package: Rphenograph (via r-universe)

November 24, 2026

Title R Implementation of the PhenoGraph Algorithm

Version 0.99.2

Description A simple R implementation of the PhenoGraph algorithm <doi:10.1016/j.cell.2015.05.047>, which is a clustering method designed for high-dimensional single-cell data analysis. It works by creating a graph representing phenotypic similarities between cells by calculating the Jaccard coefficient between nearest-neighbor sets, and then identifying communities using the Louvain method in this graph.

Depends R (>= 4.1.0)

Imports igraph (>= 2.0.0), RANN (>= 2.5), Rcpp (>= 1.0.0)

LinkingTo Rcpp

Suggests RcppHNSW, testthat, ggplot2

License Artistic-2.0

Encoding UTF-8

RoxygenNote 7.3.2

Config/pak/sysreqs libglpk-dev libxml2-dev

Repository https://peterleary.r-universe.dev

Date/Publication 2026-03-03 11:13:24 UTC

RemoteUrl https://github.com/peterleary/Rphenograph

RemoteRef HEAD

RemoteSha 0052f8b25f89e39e56ae91588e9eb84f80adf099

Contents

Rphenograph 2

Index **6**

Rphenograph *RphenoGraph clustering*

Description

R implementation of the PhenoGraph algorithm

Uses a kd-tree to find the p number of near neighbours for each point in an input/output dataset.

Usage

```
Rphenograph(
  data,
  k = 30,
  directed = FALSE,
  edge_comb = "none",
  inc_self = FALSE,
  clust_fun = NULL,
  verbose = FALSE,
  knn_fun = NULL,
  knn_report = 0
)

find_neighbors(data, k)
```

Arguments

data	matrix; input data matrix.
k	integer; number of nearest neighbours (default:30).
directed	logical; whether to use a symmetric (FALSE, default) or asymmetric ("directed", TRUE) graph.
edge_comb	character map multiple edges to single edges; it uses igraph `igraph-attribute-combination`. If graph is directed, there should be no duplicated edge. If graph is undirected, there are numerous edges that exist in both directions. Those edges will be combined using the `simplify` function, that defines which function to apply to the weights. Available values are "sum", "prod", "mean", "min", "max"... "first" is used by default as duplicated links are getting the same weight (same intersection) and is equivalent to "mean". "sum" and "prod" are reinforcing the weight of links that exist in both directions.
inc_self	logical; whether to include the 1st nearest neighbour, i.e. the data point itself, when computing the shared neighbours (intersection) with its nearest neighbours. By default, it is FALSE, as the original Rphenograph implementation. No self link is created, unless the data point is linked to none of its neighbours because they share no common neighbours.

clust_fun	function; community detection algorithm. Defaults to cluster_louvain. Louvain is not implemented for directed graphs. Other options: cluster_walktrap, cluster_spinglass, cluster_leading_eigen, cluster_edge_betweenness, cluster_fast_greedy, cluster_label_prop.
verbose	logical; verbosity (default=FALSE)
knn_fun	function or character; the function used to search the nearest neighbors. NULL points to RANN::nn2 as original code, which is configured for exact nearest neighbours (eps=0). "hnsw" points to RcppHNSW::hnsw_knn as proposed by E. Becht. Any function that returns a matrix of index to nearest neighbors could be used. NB: the first column contains the point itself and is removed before building the graph unless inc_self is TRUE.
knn_report	vector of integers; this vector defines the indices within the k nearest neighbours to report for each data points. This should be a vector of integer from 1 to k. A single value of 0 indicates that all neighbours (indices from 1 to k) will be reported, as by default.

Details

A simple R implementation of the [PhenoGraph](http://www.cell.com/cell/abstract/S0092-8674(15)00637-6) algorithm, which is a clustering method designed for high-dimensional single-cell data analysis. It works by creating a graph ("network") representing phenotypic similarities between cells by calclating the Jaccard coefficient between nearest-neighbor sets, and then identifying communities using the well known [Louvain method](https://sites.google.com/site/findcommunities/) in this graph.

The Python Phenograph algorithm defines a "prune" parameter. prune is a logical that determines whether the graph is simplified (i.e. removing reciprocal edges) by addition (FALSE, default) or multiplication (TRUE). Phenograph is using matrix operations, adjacency matrix of the graph added or multiplied to its transpose. Originally, Rphenograph does not simplify the graph, keeping reciprocal edges. Now Rphenograph offers igraph simplifications, which might lead to different results from Phenograph. In Phenograph, it's not clear what's happening when a link exists in one direction: does product lead to 0 when there is only one link?

There is no rational whether to include or not to the data point itself in the list of nearest neighbours. Phenograph does not, but the scran package does. The choice is driven here by `inc_self`. Whatever the choice, the graph could be assymmetric when a point A lists B as its k NN, but B does not, because B has k neighbours nearer than A. So A has a link to B, but B has not link to A. When A and B are in the list of each other, there are two reverse links; A and B have the same intersection and thus the same Jaccard coefficient.

Use the nn2 function from the RANN package, utilizes the Approximate Near Neighbor (ANN) C++ library, which can give the exact near neighbours or (as the name suggests) approximate near neighbours to within a specified error bound. For more information on the ANN library please visit http://www.cs.umd.edu/~mount/ANN/.

Value

a list contains an igraph graph object for `graph_from_data_frame` and a communities object, for which the most interesting operations are the following:

`membership` returns a numeric vector, one number for each vertex in the graph that was the input of the community detection.

`length` returns an integer scalar.

`sizes` returns a numeric vector.

`modularity` returns a numeric scalar.

There are also other operations:

`algorithm` returns a character scalar.

,

`crossing` returns a logical vector.

`is_hierarchical`
returns a logical scalar.

,

`merges` returns a two-column numeric matrix.

,

`cut_at` returns a numeric vector, the membership vector of the vertices.

,

`as.dendrogram` returns a dendrogram object.

,

`show_trace` returns a character vector.

,

`code_len` returns a numeric scalar for communities found with the InfoMAP method and NULL for other methods.

,

`print` returns the communities object itself, invisibly.

,

`plot` for communities objects returns NULL, invisibly.

a n-by-k matrix of neighbor indices

Functions

- `find_neighbors()`: Uses a kd-tree to find the p number of near neighbours for each point in an input/output dataset.

References

Jacob H. Levine and et.al. Data-Driven Phenotypic Dissection of AML Reveals Progenitor-like Cells that Correlate with Prognosis. Cell, 2015.

Examples

```
library(igraph)
iris_unique <- unique(iris) # Remove duplicates
data <- as.matrix(iris_unique[,1:4])
Rphenograph_out <- Rphenograph(data, k = 45)
modularity(Rphenograph_out[[2]])
membership(Rphenograph_out[[2]])
iris_unique$phenograph_cluster <- factor(membership(Rphenograph_out[[2]]))
if(require(ggplot2)) {
   ggplot(iris_unique, aes(x=Sepal.Length, y=Sepal.Width, col=Species, shape=phenograph_cluster)) +
        geom_point(size = 3) + theme_bw()
} else {
    with(iris_unique, plot(x=Sepal.Length, y=Sepal.Width,
    col=Species, pch=(15:17)[phenograph_cluster],
    cex=c(2.5, 2, 1.5)[phenograph_cluster]))
}

iris_unique <- unique(iris) # Remove duplicates
data <- as.matrix(iris_unique[,1:4])
neighbors <- find_neighbors(data, k=10)
```

Index

find_neighbors *(*Rphenograph*)*, 2

Rphenograph, 2