Package ‘graphlayouts’

March 9, 2020

Title Additional Layout Algorithms for Network Visualizations
Version 0.6.0
Maintainer David Schoch <david.schoch@manchester.ac.uk>
Description Several new layout algorithms to visualize networks are provided which are not part of ‘igraph’. Most are based on the concept of stress majorization by Gansner et al. (2004) <doi:10.1007/978-3-540-31843-9_25>. Some more specific algorithms allow to emphasize hidden group structures in networks or focus on specific nodes.

URL https://github.com/schochastics/graphlayouts
BugReports https://github.com/schochastics/graphlayouts/issues
Depends R (>= 3.2.0)
License MIT + file LICENSE
Encoding UTF-8
LazyData true
Imports igraph, Rcpp
Suggests oaqc, testthat, ggraph, ggplot2, knitr, rmarkdown
LinkingTo Rcpp, RcppArmadillo
RoxygenNote 7.0.2
NeedsCompilation yes
Author David Schoch [aut, cre]
Repository CRAN
Date/Publication 2020-03-09 15:30:06 UTC

R topics documented:

annotate_circle .......................................................... 2
draw_circle .......................................................... 3
graphlayouts .......................................................... 4
annotate_circle

graph_manipulate .......................................................... 5
layout_backbone ............................................................. 6
layout_centrality .............................................................. 7
layout_constrained_stress .................................................. 8
layout_dynamic ................................................................. 10
layout_focus ................................................................. 11
layout_manipulate ............................................................ 12
layout_pmds ................................................................. 13
layout_sparse_stress ......................................................... 14
layout_spectral .............................................................. 15
layout_stress ................................................................. 16

Index ............................................. 19

annotate_circle  annotate concentric circles

Description
annotate concentric circles

Usage
annotate_circle(cent, col = "#00BFFF", format = ", pos = "top", text_size = 3)

Arguments
cent centrality scores used for layout
col color of text
format either empty string or 'scientific'
pos position of text ('top' or 'bottom')
text_size font size for annotations

Details
this function is best used with layout_with_centrality together with draw_circle.

Value
annotated concentric circles around origin
**draw_circle**

_Draw concentric circles_

**Description**

Draw concentric circles

**Usage**

```r
draw_circle(col = "#00BFFF", use = "focus", max.circle)
```

**Arguments**

- `col`  
  color of circles

- `use`  
  one of 'focus' or 'cent'

- `max.circle`  
  if use = 'focus' specifies the number of circles to draw

**Details**

this function is best used with a concentric layout such as `layout_with_focus` and `layout_with_centrality`.

**Value**

concentric circles around origin

**Examples**

```r
library(igraph)
library(ggraph)

# sample_gnp(10, 0.4)
## Not run:
g <- sample_gnp(10, 0.4)

## Not run:
g <- sample_gnp(10, 0.4)

## Not run:
g <- sample_gnp(10, 0.4)
```

## Not run:
ggraph(g, layout = "centrality", centrality = degree(g)) +
draw_circle(use = "cent") +
geom_edge_link() +
geom_node_point(shape = 21, fill = "grey25", size = 5) +
theme_graph() +
coord_fixed()

## End(Not run)

table graphlayouts

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>The package implements several new layout algorithms to visualize networks. Most are based on the concept of stress majorization. Some more specific algorithms allow to emphasize hidden group structures in networks or focus on specific nodes. The package is best used in conjunction with ggraph.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Some features of the package are:</td>
</tr>
<tr>
<td>• layout_with_stress() is a state-of-the-art deterministic layout algorithm.</td>
</tr>
<tr>
<td>• layout_as_backbone() uncovers hidden group structures (if they exist) by emphasizing strongly embedded edges.</td>
</tr>
<tr>
<td>• layout_with_focus() and layout_with_centrality() produce concentric layouts with a focal or most central nodes in the center.</td>
</tr>
<tr>
<td>• layout_with_eigen() implements some layout algorithms on the basis of eigenvectors</td>
</tr>
<tr>
<td>• layout_with_sparse_stress() sparse stress for large graphs</td>
</tr>
<tr>
<td>• layout_with_pmds() pivot MDS for large graphs.</td>
</tr>
<tr>
<td>• layout_as_dynamic() for longitudinal network data</td>
</tr>
</tbody>
</table>

A detailed tutorial can be found here.
Description

functions to manipulate a graph

Usage

reorder_edges(g, attr, desc = TRUE)

Arguments

g           igraph object
attr         edge attribute name used to sort edges
desc         logical. sort in descending (default) or ascending order

Details

reorder_edges() allows to reorder edges according to an attribute so that edges are drawn in the given order.

Value

manipulated graph

Author(s)

David Schoch

Examples

library(igraph)
library(ggraph)

g <- sample_gnp(10, 0.5)
E(g)$attr <- 1:ecount(g)
gn <- reorder_edges(g,"attr")
**layout_backbone**

**backbone graph layout**

**Description**

emphasizes a hidden group structure if it exists in the graph. Calculates a layout for a sparsified network only including the most embedded edges. Deleted edges are added back after the layout is calculated.

**Usage**

```r
layout_as_backbone(g, keep = 0.2, backbone = TRUE)
layout_igraph_backbone(g, keep = 0.2, backbone = TRUE, circular)
```

**Arguments**

- `g`: igraph object
- `keep`: fraction of edges to keep during backbone calculation
- `backbone`: logical. Return edge ids of the backbone (Default: TRUE)
- `circular`: not used

**Details**

The `layout_igraph_*` function should not be used directly. It is only used as an argument for plotting with `igraph`. `ggraph` natively supports the layout.

**Value**

list of xy coordinates and vector of edge ids included in the backbone

**References**


**Examples**

```r
library(igraph)

g <- sample_islands(9,20,0.4,9)
g <- simplify(g)
V(g)$grp <- as.character(rep(1:9,each=20))
bb <- layout_as_backbone(g,keep=0.4)

# add backbone links as edge attribute
E(g)$col <- FALSE
```
Description

arranges nodes in concentric circles according to a centrality index.

Usage

```r
layout_with_centrality(
  g,  # igraph object
  cent,  # centrality scores
  scale = TRUE,  # logical. should centrality scores be scaled to [0, 100]? (Default: TRUE)
  iter = 500,  # number of iterations during stress optimization
  tol = 1e-04,  # stopping criterion for stress optimization
  tseq = seq(0, 1, 0.2)  # numeric vector. increasing sequence of coefficients to combine regular stress and constraint stress. See details.
)

layout_igraph_centrality(
  g,  # igraph object
  cent,  # centrality scores
  scale = TRUE,  # logical. should centrality scores be scaled to [0, 100]? (Default: TRUE)
  iter = 500,  # number of iterations during stress optimization
  tol = 1e-04,  # stopping criterion for stress optimization
  tseq = seq(0, 1, 0.2),  # numeric vector. increasing sequence of coefficients to combine regular stress and constraint stress. See details.
  circular  # not used
)
```

Arguments

g        igraph object
cent     centrality scores
scale     logical. should centrality scores be scaled to [0, 100]? (Default: TRUE)
iter     number of iterations during stress optimization
tol     stopping criterion for stress optimization
tseq     numeric vector. increasing sequence of coefficients to combine regular stress and constraint stress. See details.
circular  not used
layout_constrained_stress

Details

The function optimizes a convex combination of regular stress and a constrained stress function which forces nodes to be arranged on concentric circles. The vector \( tseq \) is the sequence of parameters used for the convex combination. In iteration \( i \) of the algorithm \( tseq[i] \) is used to combine regular and constraint stress as \( (1 - tseq[i]) \times \text{stress}_{\text{regular}} + tseq[i] \times \text{stress}_{\text{constraint}} \). The sequence must be increasing, start at zero and end at one. The default setting should be a good choice for most graphs.

The layout_igraph_* function should not be used directly. It is only used as an argument for plotting with ‘igraph’. ‘ggraph’ natively supports the layout.

Value

matrix of xy coordinates

References


Examples

```r
library(igraph)
library(ggraph)

g <- sample_gnp(10,0.4)
## Not run:
ggraph(g,layout="centrality",centrality = closeness(g))+
  draw_circle(use = "cent")+
  geom_edge_link0()+
  geom_node_point(shape = 21,fill = "grey25",size = 5)+
  theme_graph()+
  coord_fixed()
## End(Not run)
```

layout_constrained_stress

constrained stress layout

Description

force-directed graph layout based on stress majorization with variable constrained
**Usage**

```r
layout_with_constrained_stress(
  g,
  coord,
  fixdim = "x",
  weights = NA,
  iter = 500,
  tol = 1e-04,
  mds = TRUE,
  bbox = 30
)
```

```r
layout_igraph_constrained_stress(
  g,
  coord,
  fixdim = "x",
  weights = NA,
  iter = 500,
  tol = 1e-04,
  mds = TRUE,
  bbox = 30,
  circular
)
```

**Arguments**

- `g` igraph object
- `coord` numeric vector. fixed coordinates for dimension specified in `fixdim`.
- `fixdim` string. which dimension should be fixed. Either "x" or "y".
- `weights` possibly a numeric vector with edge weights. If this is NULL and the graph has a weight edge attribute, then the attribute is used. If this is NA then no weights are used (even if the graph has a weight attribute). By default, weights are ignored. See details for more.
- `iter` number of iterations during stress optimization
- `tol` stopping criterion for stress optimization
- `mds` should an MDS layout be used as initial layout (default: TRUE)
- `bbox` constrain dimension of output. Only relevant to determine the placement of disconnected graphs
- `circular` not used

**Details**

Be careful when using weights. In most cases, the inverse of the edge weights should be used to ensure that the endpoints of an edges with higher weights are closer together (weights=1/E(g)$weight).

The `layout_igraph_*` function should not be used directly. It is only used as an argument for plotting with `igraph`. `ggraph` natively supports the layout.
Value

matrix of xy coordinates

References


layout_dynamic
dynamic graph layout

Description

Create layouts for longitudinal networks.

Usage

layout_as_dynamic(gList, weights = NA, alpha = 0.5, iter = 500, tol = 1e-04)

Arguments

gList: list of igraph objects. Each network must contain the same set of nodes.
weights: possibly a numeric vector with edge weights. If this is NULL and the graph has a weight edge attribute, then the attribute is used. If this is NA then no weights are used (even if the graph has a weight attribute). By default, weights are ignored. See details for more.
alpha: weighting of reference layout. See details.
iter: number of iterations during stress optimization
tol: stopping criterion for stress optimization

Details

The reference layout is calculated based on the union of all graphs. The parameter alpha controls the influence of the reference layout. For alpha=1, only the reference layout is used and all graphs have the same layout. For alpha=0, the stress layout of each individual graph is used. Values in-between interpolate between the two layouts.

Be careful when using weights. In most cases, the inverse of the edge weights should be used to ensure that the endpoints of an edges with higher weights are closer together (weights=1/E(g)$weight).

Value

list of coordinates for each graph

References

Examples

```r
library(igraph)
g1 <- sample_gnp(20, 0.2)
g2 <- sample_gnp(20, 0.2)
g3 <- sample_gnp(20, 0.2)

xy <- layout_as_dynamic(list(g1, g2, g3))

# layout for first network
xy[[1]]
```

---

**layout_focus**  
*radial focus layout*

Description

arrange nodes in concentric circles around a focal node according to their distance from the focus.

Usage

```r
layout_with_focus(g, v, weights = NA, iter = 500, tol = 1e-04)
```

```r
layout_igraph_focus(g, v, weights = NA, iter = 500, tol = 1e-04, circular)
```

Arguments

- `g`: igraph object
- `v`: id of focal node to be placed in the center
- `weights`: possibly a numeric vector with edge weights. If this is NULL and the graph has a weight edge attribute, then the attribute is used. If this is NA then no weights are used (even if the graph has a weight attribute). By default, weights are ignored. See details for more.
- `iter`: number of iterations during stress optimization
- `tol`: stopping criterion for stress optimization
- `circular`: not used

Details

Be careful when using weights. In most cases, the inverse of the edge weights should be used to ensure that the endpoints of an edges with higher weights are closer together (weights=1/E(g)$weight).

The `layout_igraph_*` function should not be used directly. It is only used as an argument for plotting with 'igraph'. 'ggraph' natively supports the layout.

Value

a list containing xy coordinates and the distances to the focal node
References


Examples

```r
library(igraph)
library(ggraph)
g <- sample_gnp(10, 0.4)
coords <- layout_with_focus(g, v = 1)
coords
```

Description

functions to manipulate an existing layout

Usage

```r
layout_rotate(xy, angle)
layout_mirror(xy, axis = "vertical")
```

Arguments

- `xy`: graph layout
- `angle`: angle for rotation
- `axis`: mirror horizontal or vertical

Details

These functions are mostly useful for deterministic layouts such as `layout_with_stress`

Value

manipulated matrix of xy coordinates

Author(s)

David Schoch
Examples

library(igraph)
g <- sample_gnp(50,0.3)

xy <- layout_with_stress(g)

#rotate 90 degrees
xy <- layout_rotate(xy,90)

# flip horizontally
xy <- layout_mirror(xy,"horizontal")

layout_pmds

pivot MDS graph layout

Description

Similar to layout_with_mds but uses only a small set of pivots for MDS. Considerably faster than MDS and thus applicable for larger graphs.

Usage

layout_with_pmds(g, pivots, weights = NA, D = NULL)

layout_igraph_pmds(g, pivots, weights = NA, D = NULL, circular)

Arguments

g igraph object

pivots number of pivots

weights possibly a numeric vector with edge weights. If this is NULL and the graph has a weight edge attribute, then the attribute is used. If this is NA then no weights are used (even if the graph has a weight attribute). By default, weights are ignored. See details for more.

D precomputed distances from pivots to all nodes (if available, default: NULL)

circular not used

Details

Be careful when using weights. In most cases, the inverse of the edge weights should be used to ensure that the endpoints of an edges with higher weights are closer together (weights=1/E(g)$weight)

The layout_igraph_* function should not be used directly. It is only used as an argument for plotting with ’igraph’. ’ggraph’ natively supports the layout.
layout_sparse_stress

Value

matrix of xy coordinates

Author(s)

David Schoch

References


Examples

```r
## Not run:
library(igraph)
library(ggraph)

g <- sample_gnp(1000, 0.01)
xy <- layout_with_pmds(g, pivots = 100)
## End(Not run)
```

layout_sparse_stress  
*sparse stress graph layout*

Description

stress majorization for larger graphs based on a set of pivot nodes.

Usage

```r
layout_with_sparse_stress(g, pivots, weights = NA, iter = 500)

layout_igraph_sparse_stress(g, pivots, weights = NA, iter = 500, circular)
```

Arguments

- `g`  
  igraph object
- `pivots`  
  number of pivots
- `weights`  
  ignored
- `iter`  
  number of iterations during stress optimization
- `circular`  
  not used
**layout_spectral**

**Details**

The `layout_igraph_*` function should not be used directly. It is only used as an argument for plotting with `igraph`. `ggraph` natively supports the layout.

**Value**

matrix of xy coordinates

**Author(s)**

David Schoch

**References**


**Examples**

```r
## Not run:
library(igraph)
library(ggraph)

g <- sample_gnp(1000, 0.005)

ggraph(g, layout = "sparse_stress", pivots = 100) +
  geom_edge_link0(edge_colour = "grey66") +
  geom_node_point(shape = 21, fill = "grey25", size = 5) +
  theme_graph()

## End(Not run)
```

---

**layout_spectral**  
*spectral graph layouts*

**Description**

Using a set of eigenvectors of matrices associated with a graph as coordinates

**Usage**

```r
layout_with_eigen(g, type = "laplacian", ev = "smallest")

layout_igraph_eigen(g, type = "laplacian", ev = "smallest", circular)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>g</code></td>
<td>igraph object</td>
</tr>
<tr>
<td><code>type</code></td>
<td>matrix to be used for spectral decomposition. either 'adjacency' or 'laplacian'</td>
</tr>
<tr>
<td><code>ev</code></td>
<td>eigenvectors to be used. Either 'smallest' or 'largest'.</td>
</tr>
<tr>
<td><code>circular</code></td>
<td>not used</td>
</tr>
</tbody>
</table>
Details

The `layout_igraph_*` function should not be used directly. It is only used as an argument for plotting with `igraph`. `ggraph` natively supports the layout.

Value

matrix of xy coordinates

Author(s)

David Schoch

Examples

```r
library(igraph)

g <- sample_gnp(50, 0.2)

xy <- layout_with_eigen(g, type = "adjacency", ev = "largest")

xy <- layout_with_eigen(g, type = "adjacency", ev = "smallest")

xy <- layout_with_eigen(g, type = "laplacian", ev = "largest")

xy <- layout_with_eigen(g, type = "laplacian", ev = "smallest")
```

---

`layout_stress`  
stress majorization layout

Description

force-directed graph layout based on stress majorization.

Usage

```r
layout_with_stress(

  g,
  weights = NA,
  iter = 500,
  tol = 1e-04,
  mds = TRUE,
  bbox = 30
)

layout_igraph_stress(

  g,
  weights = NA,
  iter = 500,
```
layout_stress

tol = 1e-04,
mds = TRUE,
bbox = 30,
circular
)

Arguments

- **g**: igraph object
- **weights**: possibly a numeric vector with edge weights. If this is NULL and the graph has a weight edge attribute, then the attribute is used. If this is NA then no weights are used (even if the graph has a weight attribute). By default, weights are ignored. See details for more.
- **iter**: number of iterations during stress optimization
- **tol**: stopping criterion for stress optimization
- **mds**: should an MDS layout be used as initial layout (default: TRUE)
- **bbox**: constrain dimension of output. Only relevant to determine the placement of disconnected graphs
- **circular**: not used

Details

Be careful when using weights. In most cases, the inverse of the edge weights should be used to ensure that the endpoints of an edges with higher weights are closer together (weights=1/E(g)$weight).

The `layout_igraph_*` function should not be used directly. It is only used as an argument for plotting with `igraph`. `ggraph` natively supports the layout.

Value

matrix of xy coordinates

References


Examples

```r
library(igraph)
library(ggraph)
set.seed(665)

# calculate layout manually
xy <- layout_with_stress(g)

# use it with ggraph
```
## Not run:
```r
ggraph(g, layout = "stress") +
  geom_edge_link(edge_width = 0.2, colour = "grey") +
  geom_node_point(col = "black", size = 0.3) +
  theme_graph()
```
## End(Not run)
Index

annotate_circle, 2

draw_circle, 2, 3

graph_manipulate, 5

graphlayouts, 4

layout_as_backbone (layout_backbone), 6
layout_as_dynamic (layout_dynamic), 10
layout_backbone, 6
layout_centrality, 7
layout_constrained_stress, 8
layout_dynamic, 10
layout_focus, 11
layout_igraph_backbone
  (layout_backbone), 6
layout_igraph_centrality
  (layout_centrality), 7
layout_igraph_constrained_stress
  (layout_constrained_stress), 8
layout_igraph_eigen (layout_spectral), 15
layout_igraph_focus (layout_focus), 11
layout_igraph_pmds (layout_pmds), 13
layout_igraph_sparse_stress
  (layout_sparse_stress), 14
layout_igraph_stress (layout_stress), 16
layout_manipulate, 12
layout_mirror (layout_manipulate), 12
layout_pmds, 13
layout_rotate (layout_manipulate), 12
layout_sparse_stress, 14
layout_spectral, 15
layout_stress, 16
layout_with_centrality, 2, 3
layout_with_centrality
  (layout_centrality), 7
layout_with_constrained_stress
  (layout_constrained_stress), 8
layout_with_eigen (layout_spectral), 15
layout_with_focus, 3
layout_with_focus (layout_focus), 11
layout_with_mds, 13
layout_with_pmds (layout_pmds), 13
layout_with_sparse_stress
  (layout_sparse_stress), 14
layout_with_stress, 12
layout_with_stress (layout_stress), 16
reorder_edges (graph_manipulate), 5