Package ‘snowboot’

April 25, 2020

Type Package

Title Bootstrap Methods for Network Inference

Version 1.0.2

Date 2020-04-23

Description Functions for analysis of network objects, which are imported or simulated by the package. The non-parametric methods of analysis center on snowball and bootstrap sampling for estimating functions of network degree distribution. For other parameters of interest, see, e.g., 'bootnet' package.

Depends R (>= 3.3.0)

License GPL-3

LazyData TRUE

Imports graphics, igraph, parallel, Rcpp, Rdpack, stats

RdMacros Rdpack

RoxygenNote 7.1.0

LinkingTo Rcpp

Encoding UTF-8

NeedsCompilation yes

Author Leticia Ramirez-Ramirez [aut],
Kusha Nezafati [aut],
Yuzhou Chen [aut],
Vyacheslav Lyubchich [aut, cre]
(<https://orcid.org/0000-0001-7936-4285>),
Yulia R. Gel [aut]

Maintainer Vyacheslav Lyubchich <lyubchich@umces.edu>

Repository CRAN

Date/Publication 2020-04-25 18:30:03 UTC

R topics documented:

`artificial_networks` *10 Simulated Networks of Order 2000 with Polylogarithmic (0.1, 2) Degree Distributions*

Description

A list called "artificial_networks". The length of the list is 10, and each element is a network object of order 2000. These networks were simulated using the polylogarithmic (aka Gutenberg–Richter law) degree distribution with parameters $\delta = 0.1$ and $\lambda = 2$ as shown in the following equations:

$$f(k) = k^{-\delta} e^{-k/\lambda} / Li_\delta(e^{-1/\lambda})$$

$$Li_\delta(z) = \sum_{j=1}^{\infty} z^{-j} / j^\delta,$$

where $\lambda > 0$ (see Newman et al. 2001, Gel et al. 2017, and Chen et al. 2018 for details).

Usage

```
artificial_networks
```

Format

A list containing 10 network objects. Each network object is a list with three elements:

`degree` the degree sequence of the network, which is an integer vector of length n;

`edges` the edgelist, which is a two-column matrix, where each row is an edge of the network;

`n` the network order (number of nodes in the network). The order is 2000.

References

Chen Y, Gel YR, Lyubchich V, Nezafati K (2018). "Snowboot: bootstrap methods for network inference." *The R Journal*, **10**(2), 95–113. doi: 10.32614/RJ2018056.

Gel YR, Lyubchich V, Ramirez Ramirez LL (2017). "Bootstrap quantification of estimation uncertainties in network degree distributions." *Scientific Reports*, **7**, 5807. doi: 10.1038/s41598017-05885x.

Newman MEJ, Strogatz SH, Watts DJ (2001). "Random graphs with arbitrary degree distributions and their applications." *Physical Review E*, **64**(2), 026118. doi: 10.1103/PhysRevE.64.026118.

`boot_ci` *Confidence Intervals from Bootstrapped Network Degree Distribution*

Description

The function calculates bootstrap confidence intervals for the parameters of network degree distribution: probabilities of node degrees $f(k)$ and mean degree μ, where $k = 0, 1, \ldots$ are the degrees.

Usage

```
boot_ci(x, prob = 0.95, method = c("percentile", "basic"))
```

Arguments

`x`	a list with bootstrapped results – output of `boot_dd`.
`prob`	confidence level for the intervals. Default is 0.95 (i.e., 95% confidence).
`method`	method for calculating the bootstrap intervals. Default is `"percentile"` (see Details).

Details

Currently, the bootstrap intervals can be calculated with two alternative methods: `"percentile"` or `"basic"`. The `"percentile"` intervals correspond to Efron's $100\cdot$`prob`% intervals (see Efron 1979, also Equation 5.18 by Davison and Hinkley 1997 and Equation 3 by Gel et al. 2017, Chen et al. 2018):

$$(\theta^*_{[B\alpha/2]}, \theta^*_{[B(1-\alpha/2)]}),$$

where $\theta^*_{[B\alpha/2]}$ and $\theta^*_{[B(1-\alpha/2)]}$ are empirical quantiles of the bootstrap distribution with `B` bootstrap replications for parameter θ (θ can be the $f(k)$ or μ), and $\alpha = 1-$ `prob`.

The `"basic"` method produces intervals (see Equation 5.6 by Davison and Hinkley 1997):

$$(2\hat{\theta} - \theta^*_{[B(1-\alpha/2)]}, 2\hat{\theta} - \theta^*_{[B\alpha/2]}),$$

where $\hat{\theta}$ is the sample estimate of the parameter. Note that this method can lead to negative confidence bounds, especially when $\hat{\theta}$ is close to 0.

Value

A list object of class "snowboot" with the following elements:

- `fk_ci` A matrix of dimensions $2\times$`length(x$fk)`, where the number of columns corresponds to the number of probabilities $f(k)$ estimated from an LSMI sample. Each column of the matrix is a confidence interval for a corresponding $f(k)$. I.e., the first row of the matrix gives the lower bounds, while the second row contains all upper bounds.
- `mu_ci` A numeric vector of length 2 with lower and upper confidence bounds for the network mean degree μ.
- `prob` Confidence level for the intervals.
- `method` Method that was used for calculating the bootstrap intervals.
- `fk` A vector with an estimate of the degree distribution, copied from the input `x$fk`.
- `mu` An estimate of the mean degree, copied from the input `x$mu`.

References

Chen Y, Gel YR, Lyubchich V, Nezafati K (2018). "Snowboot: bootstrap methods for network inference." *The R Journal*, **10**(2), 95–113. doi: 10.32614/RJ2018056.

Davison AC, Hinkley DV (1997). *Bootstrap Methods and Their Application*. Cambridge University Press, Cambridge.

Efron B (1979). "Bootstrap methods: Another look at the jackknife." *The Annals of Statistics*, **7**(1), 1–26. doi: 10.1214/aos/1176344552.

Gel YR, Lyubchich V, Ramirez Ramirez LL (2017). "Bootstrap quantification of estimation uncertainties in network degree distributions." *Scientific Reports*, **7**, 5807. doi: 10.1038/s41598017-05885x.

See Also

boot_dd

Examples

```
net <- artificial_networks[[1]]
lsmiEstimate <- lsmi_dd(net = net, n.seed = 5, n.wave = 3)
bootEstimates <- boot_dd(lsmiEstimate, B = 10)
bootIntervals1 <- boot_ci(bootEstimates)

#Another version of the intervals:
bootIntervals2 <- boot_ci(bootEstimates, method = "basic")
```

boot_dd *Bootstrapping Empirical Degree Distribution*

Description

This function delivers bootstrap estimates of network degree distribution based on an LSMI sample. The bootstrap scheme is non-weighted for seeds (resampling with replacement) and weighted for non-seeds (resampling with replacement, with weights proportional to inverse of the degrees), as described in Section 3.3 by Thompson et al. (2016) and in Algorithm 1 by Gel et al. (2017).

Usage

```
boot_dd(x, B = 100, cl = 1)
```

Arguments

x	a list that is the output of `lsmi_dd`, i.e., an estimate of the degree distribution together with all degrees of seeds and non-seeds from an LSMI.
B	a positive integer, the number of bootstrap replications to perform. Default is 100.
cl	parameter to specify computer cluster for bootstrapping, passed to the package `parallel` (default is `1`, meaning no cluster is used). Possible values are: • cluster object (list) produced by makeCluster. In this case, new cluster is not started nor stopped; • `NULL`. In this case, the function will attempt to detect available cores (see detectCores) and, if there are multiple cores (>1), a cluster will be started with makeCluster. If started, the cluster will be stopped after computations are finished; • positive integer defining the number of cores to start a cluster. If `cl = 1`, no attempt to create a cluster will be made. If `cl > 1`, cluster will be started (using makeCluster) and stopped afterwards (using stopCluster).

Value

A list object of class `"snowboot"` consisting of:

fkb	A matrix of dimensions `length(x$fk)`$\times$`B` with `B` bootstrap estimates of the degree distribution. The bootstrap estimates are computed according to Equation 1 by Gel et al. (2017), also see Chen et al. (2018).
mub	A vector of length `B` with bootstrapped estimates of the network mean degree. The bootstrap estimates are computed according to Equation 2 by Gel et al. (2017).
fk	A vector with an estimate of the degree distribution, copied from the input `x$fk`.
mu	An estimate of the mean degree, copied from the input `x$mu`.
B	The number of bootstrap replications performed.

References

Chen Y, Gel YR, Lyubchich V, Nezafati K (2018). "Snowboot: bootstrap methods for network inference." *The R Journal*, **10**(2), 95–113. doi: 10.32614/RJ2018056.

Gel YR, Lyubchich V, Ramirez Ramirez LL (2017). "Bootstrap quantification of estimation uncertainties in network degree distributions." *Scientific Reports*, **7**, 5807. doi: 10.1038/s41598017-05885x.

Thompson ME, Ramirez Ramirez LL, Lyubchich V, Gel YR (2016). "Using the bootstrap for statistical inference on random graphs." *Canadian Journal of Statistics*, **44**(1), 3–24. doi: 10.1002/cjs.11271.

See Also

`lsmi`, `lsmi_dd`, `boot_ci`

Examples

```
net <- artificial_networks[[1]]
lsmiEstimate <- lsmi_dd(net = net, n.seed = 5, n.wave = 3)
bootEstimates <- boot_dd(lsmiEstimate, B = 10)
```

`igraph_to_network` *Create a "Network" Object from an igraph Object*

Description

This function converts an igraph object to an object compatible with snowboot functions.

Usage

```
igraph_to_network(in_graph)
```

Arguments

`in_graph`	An igraph object. To create igraph objects from field data, see `graph_from_edgelist`, `graph_from_data_frame`, `graph_from_adjacency_matrix`, or `read_graph`.

Value

A list that contain elements:

`edges`	The edgelist of the network. A two column `matrix` where each row is an edge.
`degree`	The degree sequence of the network, which is an `integer` vector of length n.
`n`	The network order.

References

http://igraph.org/

Examples

```
hex_ring <- igraph::make_ring(6, directed = FALSE, mutual = FALSE, circular = TRUE)
net <- igraph_to_network(hex_ring)
```

---

lsmi — *Labeled Snowball with Multiple Inclusions (LSMI)*

Description

Obtain LSMI samples around several seeds, which can be selected randomly or pre-specified. See Figure 1 by Gel et al. (2017) or Figure 2 by Chen et al. (2018) illustrating the algorithm of sampling around multiple seeds.

Usage

```
lsmi(net, n.seed = 10, n.wave = 1, seeds = NULL)
```

Arguments

`net` a network object that is a list containing:

- `degree` the degree sequence of the network, which is an `integer` vector of length n;
- `edges` the edgelist, which is a two-column matrix, where each row is an edge of the network;
- `n` the network order (i.e., number of nodes in the network).

The network object can be simulated by `random_network`, selected from the networks available in `artificial_networks`, converged from an `igraph` object with `igraph_to_network`, etc.

`n.seed` an integer defining the number of nodes to randomly sample from the network to start an LSMI sample around each of them.

`n.wave` an integer defining the number of waves (order of the neighborhood) to be recorded around the seed in the LSMI. For example, `n.wave = 1` corresponds to an LSMI with the seed and its first neighbors. Note that the algorithm allows for multiple inclusions.

`seeds` a vector of numeric IDs of pre-specified seeds. If specified, LSMIs are constructed around each such seed.

Details

If `seeds` specified, `n.seed` is not used.

Value

A list of length `n.seed` (or, if `seeds` are specified, of length `length(unique(seeds))`), where each element is a list of length `n.wave + 1` representing an LSMI produced by `sample_about_one_seed`.

References

Chen Y, Gel YR, Lyubchich V, Nezafati K (2018). "Snowboot: bootstrap methods for network inference." *The R Journal*, **10**(2), 95–113. doi: 10.32614/RJ2018056.

Gel YR, Lyubchich V, Ramirez Ramirez LL (2017). "Bootstrap quantification of estimation uncertainties in network degree distributions." *Scientific Reports*, **7**, 5807. doi: 10.1038/s41598017-05885x.

See Also

`sample_about_one_seed`, `lsmi_union`

Examples

```
net <- artificial_networks[[1]]
a <- lsmi(net, n.seed = 20, n.wave = 2)
```

---

lsmi_cv — *Cross-validation to Select an Optimal Combination of n.seed and n.wave*

Description

From the vector of specified `n.seeds` and possible waves `1:n.wave` around each seed, the function selects a single number `n.seed` and an `n.wave` (optimal seed-wave combination) that produce a labeled snowball with multiple inclusions (LSMI) sample with desired bootstrap confidence intervals for a parameter of interest. Here by 'desired' we mean that the interval (and corresponding seed-wave combination) are selected as having the best coverage (closest to the specified level `prob`), based on a cross-validation procedure with proxy estimates of the parameter. See Algorithm 2 by Gel et al. (2017) and Details below.

Usage

```
lsmi_cv(
  net,
  n.seeds,
  n.wave,
  seeds = NULL,
  B = 100,
  prob = 0.95,
  cl = 1,
```

```
  param = c("mu"),
  method = c("percentile", "basic"),
  proxyRep = 19,
  proxySize = 30
)
```

Arguments

- `net` a network object that is a list containing:
  - `degree` the degree sequence of the network, which is an `integer` vector of length n;
  - `edges` the edgelist, which is a two-column matrix, where each row is an edge of the network;
  - `n` the network order (i.e., number of nodes in the network).

  The network object can be simulated by `random_network`, selected from the networks available in `artificial_networks`, converged from an `igraph` object with `igraph_to_network`, etc.
- `n.seeds` an integer vector of numbers of seeds for snowball sampling (cf. a single integer `n.seed` in `lsmi`). Only `n.seeds` <= `n` are retained. If `seeds` is specified, only values `n.seeds` < `length(unique(seeds))` are retained and automatically supplemented by `length(unique(seeds))`.
- `n.wave` an integer defining the number of waves (order of the neighborhood) to be recorded around the seed in the LSMI. For example, `n.wave = 1` corresponds to an LSMI with the seed and its first neighbors. Note that the algorithm allows for multiple inclusions.
- `seeds` a vector of numeric IDs of pre-specified seeds. If specified, LSMIs are constructed around each such seed.
- `B` a positive integer, the number of bootstrap replications to perform. Default is 100.
- `prob` confidence level for the intervals. Default is 0.95 (i.e., 95% confidence).
- `cl` parameter to specify computer cluster for bootstrapping, passed to the package `parallel` (default is `1`, meaning no cluster is used). Possible values are:
  - cluster object (list) produced by makeCluster. In this case, new cluster is not started nor stopped;
  - `NULL`. In this case, the function will attempt to detect available cores (see detectCores) and, if there are multiple cores (> 1), a cluster will be started with makeCluster. If started, the cluster will be stopped after computations are finished;
  - positive integer defining the number of cores to start a cluster. If `cl = 1`, no attempt to create a cluster will be made. If `cl > 1`, cluster will be started (using makeCluster) and stopped afterwards (using stopCluster).
- `param` The parameter of interest for which to run a cross-validation and select optimal `n.seed` and `n.wave`. Currently, only one selection is possible: `"mu"` (the network mean degree).

`method`	method for calculating the bootstrap intervals. Default is `"percentile"` (see Details).
`proxyRep`	The number of times to repeat proxy sampling. Default is 19.
`proxySize`	The size of the proxy sample. Default is 30.

Details

Currently, the bootstrap intervals can be calculated with two alternative methods: `"percentile"` or `"basic"`. The `"percentile"` intervals correspond to Efron's $100 \cdot$prob% intervals (see Efron 1979, also Equation 5.18 by Davison and Hinkley 1997 and Equation 3 by Gel et al. 2017, Chen et al. 2018):

$$(\theta^*_{[B\alpha/2]}, \theta^*_{[B(1-\alpha/2)]}),$$

where $\theta^*_{[B\alpha/2]}$ and $\theta^*_{[B(1-\alpha/2)]}$ are empirical quantiles of the bootstrap distribution with `B` bootstrap replications for parameter θ (θ can be the $f(k)$ or μ), and $\alpha = 1-$ `prob`.

The `"basic"` method produces intervals (see Equation 5.6 by Davison and Hinkley 1997):

$$(2\hat{\theta} - \theta^*_{[B(1-\alpha/2)]}, 2\hat{\theta} - \theta^*_{[B\alpha/2]}),$$

where $\hat{\theta}$ is the sample estimate of the parameter. Note that this method can lead to negative confidence bounds, especially when $\hat{\theta}$ is close to 0.

Value

A list consisting of:

`bci`	A numeric vector of length 2 with the bootstrap confidence interval (lower bound, upper bound) for the parameter of interest. This interval is obtained by bootstrapping node degrees in an LSMI with the optimal combination of `n.seed` and `n.wave` (the combination is reported in `best_combination`).
`estimate`	Point estimate of the parameter of interest (based on the LSMI with `n.seed` seeds and `n.wave` waves reported in the `best_combination`).
`best_combination`	An integer vector of lenght 2 containing the optimal `n.seed` and `n.wave` selected via cross-validation.
`seeds`	A vector of numeric IDs of the seeds that were used in the LSMI with the optimal combination of `n.seed` and `n.wave`.

References

Chen Y, Gel YR, Lyubchich V, Nezafati K (2018). "Snowboot: bootstrap methods for network inference." *The R Journal*, **10**(2), 95–113. doi: 10.32614/RJ2018056.

Davison AC, Hinkley DV (1997). *Bootstrap Methods and Their Application*. Cambridge University Press, Cambridge.

Efron B (1979). "Bootstrap methods: Another look at the jackknife." *The Annals of Statistics*, **7**(1), 1–26. doi: 10.1214/aos/1176344552.

Gel YR, Lyubchich V, Ramirez Ramirez LL (2017). "Bootstrap quantification of estimation uncertainties in network degree distributions." *Scientific Reports*, **7**, 5807. doi: 10.1038/s41598017-05885x.

See Also

lsmi, lsmi_union, boot_dd, boot_ci

Examples

```
net <- artificial_networks[[1]]
a <- lsmi_cv(net, n.seeds = c(10, 20, 30), n.wave = 5, B = 100)
```

---

lsmi_dd — *Network Degree Distribution Estimated from Labeled Snowball Sample with Multiple Inclusion (LSMI)*

Description

lsmi_dd computes an empirical network degree distribution and estimates mean degree based on data from an LSMI sample from a network; see Equations 6 and 7 by Thompson et al. (2016) and Equation 1 by Chen et al. (2018) on the details of the calculations.

Usage

```
lsmi_dd(x = NULL, net, ...)
```

Arguments

- `x` — the LSMI sample obtained from the network `net`, for example, with `lsmi` function or as a subset of the output by `lsmi_union`.
- `net` — a network object that is a list containing:
  - `degree` the degree sequence of the network, which is an `integer` vector of length n;
  - `edges` the edgelist, which is a two-column matrix, where each row is an edge of the network;
  - `n` the network order (i.e., number of nodes in the network).

  The network object can be simulated by `random_network`, selected from the networks available in `artificial_networks`, converged from an `igraph` object with `igraph_to_network`, etc.
- `...` — arguments passed to the `lsmi` function (ignored if `x` is specified, see Details).

Details

The samples produced with lsmi or lsmi_union contain just node IDs arranged into lists of seeds and waves (no details on the node degrees or other node features). This information is sufficient to study some properties of a network (e.g., network motifs – not yet implemented in the package). To estimate a degree distribution or mean degree, both the LSMI sample and the original network object are required. If the LSMI object x is not supplied, the function will attempt sampling an LSMI automatically, using the arguments supplied in "..." that will be passed to the lsmi function.

Value

A list object of class "snowboot" consisting of:

- `fk` — A named numeric vector with estimated probabilities $\hat{f}(k)$ of degrees k, where $k = 0, 1, \ldots,$ max(c(ds,dns)) (i.e., k ranges from 0 to the maximum node degree observed in the LSMI sample). The names of the vector elements are k.
- `mu` — An estimate of the mean degree.
- `ds` — An integer vector of degrees of seed nodes.
- `dns` — An integer vector of degrees of non-seed nodes (i.e., nodes recorded in the waves of neighbors).

References

Chen Y, Gel YR, Lyubchich V, Nezafati K (2018). "Snowboot: bootstrap methods for network inference." *The R Journal*, **10**(2), 95–113. doi: 10.32614/RJ2018056.

Thompson ME, Ramirez Ramirez LL, Lyubchich V, Gel YR (2016). "Using the bootstrap for statistical inference on random graphs." *Canadian Journal of Statistics*, **44**(1), 3–24. doi: 10.1002/cjs.11271.

See Also

lsmi, lsmi_union, boot_dd

Examples

```
net <- artificial_networks[[1]]

#Obtain an LSMI sample and, at the next step,
#use it to estimate the degree distribution:
lsmiSample <- lsmi(net, n.seed = 5, n.wave = 3)
fkEstimate1 <- lsmi_dd(lsmiSample, net)$fk

#Obtain an LSMI sample and estimate the degree
#distribution in a single step:
fkEstimate2 <- lsmi_dd(net = net, n.seed = 5, n.wave = 3)$fk

#Use the output of lsmi_union to get the estimate:
lsmiUnionSample <- lsmi_union(net, n.seeds = c(5, 10), n.wave = 3)
fkEstimate3 <- lsmi_dd(lsmiUnionSample$lsmi_big, net)$fk
```

`lsmi_union` *Snowball Sampling with Multiple Inclusions around Several Subsets of Seeds*

Description

Obtain one big LSMI – with `max(n.seeds)` seeds and `n.wave` waves around each – and subsample seeds to create smaller LSMIs (with less seeds and/or waves). The function is primarily used in cross-validation.

Usage

```
lsmi_union(net, n.seeds, n.wave, seeds = NULL)
```

Arguments

`net`	a network object that is a list containing: `degree` the degree sequence of the network, which is an `integer` vector of length n; `edges` the edgelist, which is a two-column matrix, where each row is an edge of the network; `n` the network order (i.e., number of nodes in the network). The network object can be simulated by `random_network`, selected from the networks available in `artificial_networks`, converged from an `igraph` object with `igraph_to_network`, etc.
`n.seeds`	an integer vector of numbers of seeds for snowball sampling (cf. a single integer `n.seed` in `lsmi`). Only `n.seeds` <= `n` are retained. If `seeds` is specified, only values `n.seeds` < `length(unique(seeds))` are retained and automatically supplemented by `length(unique(seeds))`.
`n.wave`	an integer defining the number of waves (order of the neighborhood) to be recorded around the seed in the LSMI. For example, `n.wave = 1` corresponds to an LSMI with the seed and its first neighbors. Note that the algorithm allows for multiple inclusions.
`seeds`	a vector of numeric IDs of pre-specified seeds. If specified, LSMIs are constructed around each such seed.

Details

Note that the produced LSMIs are slightly different from those described by Gel et al. (2017). The current R implementation produces smaller LSMIs by subsetting the seeds, not by new sampling of seeds from the network and growing completely new LSMIs, as it was done by Gel et al. (2017). See the details in Figure 3 by Chen et al. (2018)

Value

A list with two elements:

- `lsmi_big` LSMI with `max(n.seeds)` seeds (see the argument definition above) and `n.wave` waves produced by the `lsmi` function.
- `sequence_seeds` A list of length equal to `length(n.seeds)`; each element of the list is a random subset of the seeds' IDs, starting from the largest (a set of size `max(n.seeds)`) to the smallest (a set of size `min(n.seeds)`).

References

Chen Y, Gel YR, Lyubchich V, Nezafati K (2018). "Snowboot: bootstrap methods for network inference." *The R Journal*, **10**(2), 95–113. doi: 10.32614/RJ2018056.

Gel YR, Lyubchich V, Ramirez Ramirez LL (2017). "Bootstrap quantification of estimation uncertainties in network degree distributions." *Scientific Reports*, **7**, 5807. doi: 10.1038/s41598017-05885x.

See Also

`sample_about_one_seed`, `lsmi`, `lsmi_cv`

Examples

```
net <- artificial_networks[[1]]
a <- lsmi_union(net, n.seeds = c(5, 10, 15), n.wave = 2)
```

`plot.snowboot` *Plot Degree Distribution Estimates*

Description

Plot LSMI-based point estimates of probabilities of node degrees, $\hat{f}(k)$, and of mean degree, $\hat{\mu}$, where $k = 0, 1, \ldots$ are the degrees. The point estimates are supplemented with box-and-whisker plots of bootstrapped values (if the input is a `boot_dd` output) or element-wise bootstrap confidence intervals (if the input is a `boot_ci` output). See Chen et al. (2018).

Usage

```
## S3 method for class 'snowboot'
plot(
  x,
  k = NULL,
  plotmu = TRUE,
  plotlegend = TRUE,
  col0 = "gray50",
```

```
  lwd0 = 1,
  colpt = "royalblue3",
  lwdpt = 2,
  pchpt = 4,
  coli = "palegreen3",
  colibg = "palegreen",
  length = 0.1,
  boxwex = 0.4,
  legendargs = list(x = "topright", cex = 0.9, bty = "n"),
  las = 1,
  ...
)
```

Arguments

`x`	output of `lsmi_dd`, `boot_dd`, or `boot_ci`.
`k`	an integer vector with degrees to plot. By default, all degrees represented in `x` are plotted.
`plotmu`	logical value indicating whether to plot the results for mean degree (default is `TRUE`).
`plotlegend`	logical value indicating whether to plot a legend (default is `TRUE`).
`col0`	color to plot horizontal zero-line at $f(k) = 0$. Use `NA` for no plotting.
`lwd0`	width of the horizontal zero-line at $f(k) = 0$.
`colpt`	color for plotting point estimates.
`lwdpt`	line width for plotting point estimates.
`pchpt`	point type for plotting point estimates (see argument pch in `points`).
`coli`	color for plotting lines or borders of box-plots for bootstrap estimates.
`colibg`	background color, if plotting boxplots of bootstrapped estimates (see argument `border` in `boxplot`).
`length`	length of arrows, if plotting bootstrap confidence intervals (see argument `length` in `arrows`).
`boxwex`	argument of `boxplot` function.
`legendargs`	additional arguments for plotting the legend (see arguments in `legend`).
`las`	argument of `plot` function.
`...`	additional arguments to pass to the `plot` function.

References

Chen Y, Gel YR, Lyubchich V, Nezafati K (2018). "Snowboot: bootstrap methods for network inference." *The R Journal*, **10**(2), 95–113. doi: 10.32614/RJ2018056.

Examples

```
net <- artificial_networks[[1]]
x <- lsmi_dd(net = net, n.wave = 2, n.seed = 40)
plot(x)

x2 <- boot_dd(x)
plot(x2, k = c(1:10))

x3 <- boot_ci(x2, prob = 0.99)
plot(x3, k = c(1:10))
```

---

random_network *Construct Artificial Networks*

Description

This function constructs an artificial network from a given distribution. Only 11 distributions are available.

Usage

```
random_network(n, distrib, param = NULL, degree = NULL, take.p = 0.05)
```

Arguments

n	the number of nodes in the desired network.
distrib	an atomic character representing the desired degree distribution. User may choose from 11 available distributions: "fixed", "pois", "ztpois", "geom", "nbinom", "ztgeom", "poly.log", "logarithmic", "power.law", "full" (fully connected), or "none" (no element connected).
param	the distribution parameters. If the function is "fixed", `param` is a vector of degrees.
degree	an optional vector of degrees that must be of length `n`. The default is `degree = NULL`.
take.p	a number between 0 and 1 representing the proportion to take for elimination with each iteration.

Value

A list consisting of:

edges	The edgelist of the network. A two column `matrix` where each row is an edge.
degree	The degree sequence of the network, which is an `integer` vector of length n.
degree.left	A vector of length `n` that should be all zeroes.
n	The network order. The order for every network is 2000.

Examples

```
a <- random_network(1000, "poly.log", c(2, 13))
```

`sample_about_one_seed` *Snowball Sampling with Multiple Inclusions around One Network Node*

Description

This function obtains a labeled snowball with multiple inclusions (LSMI) sample, starting from a single network node called seed. See Figure 1 by Thompson et al. (2016) illustrating the algorithm of sampling around one seed.

Usage

```
sample_about_one_seed(net, seed, n.wave = 1)
```

Arguments

`net`	a network object that is a list containing: `degree` the degree sequence of the network, which is an `integer` vector of length n; `edges` the edgelist, which is a two-column matrix, where each row is an edge of the network; `n` the network order (i.e., number of nodes in the network). The network object can be simulated by `random_network`, selected from the networks available in `artificial_networks`, converged from an `igraph` object with `igraph_to_network`, etc.
`seed`	numeric ID of a seed to start the LSMI.
`n.wave`	an integer defining the number of waves (order of the neighborhood) to be recorded around the seed in the LSMI. For example, `n.wave = 1` corresponds to an LSMI with the seed and its first neighbors. Note that the algorithm allows for multiple inclusions.

Value

`sample_about_one_seed` returns a list of length `n.wave + 1` containing ID of the seed (1st element of the output list), IDs of nodes in the 1st wave (2nd element of the list), . . . , IDs of nodes in the wave `n.wave` ((`n.wave + 1`)th element of the list). If a wave has no nodes in it, the corresponding element of the output contains `NA`.

References

Thompson ME, Ramirez Ramirez LL, Lyubchich V, Gel YR (2016). "Using the bootstrap for statistical inference on random graphs." *Canadian Journal of Statistics*, **44**(1), 3–24. doi: 10.1002/cjs.11271.

See Also

lsmi

Examples

```
net <- artificial_networks[[1]]
a <- sample_about_one_seed(net, seed = 1, n.wave = 2)
```

---

vertboot *Bootstrapping a Network with Vertex Bootstrap*

Description

This function bootstraps the original network using a vertex bootstrap technique. See Snijders and Borgatti (1999) and Chen et al. (2018).

Usage

```
vertboot(m1, boot_rep)
```

Arguments

m1	An adjacency matrix of the original network.
boot_rep	A positive integer number, the number of bootstrap replications.

Value

A list of bootstrapped networks as adjacency matrices.

References

Chen Y, Gel YR, Lyubchich V, Nezafati K (2018). "Snowboot: bootstrap methods for network inference." *The R Journal*, **10**(2), 95–113. doi: 10.32614/RJ2018056.

Snijders TAB, Borgatti SP (1999). "Non-parametric standard errors and tests for network statistics." *Connections*, **22**(2), 61–70.

Examples

```
graph_ex <- igraph::graph_from_edgelist(artificial_networks[[1]]$edges)
m1 <- igraph::as_adjacency_matrix(graph_ex)
m1 <- as.matrix(m1)
vertboot_out <- vertboot(m1, 20)
```

Index