Package: spreadr (via r-universe)

November 3, 2026

Type Package

Title Simulating Spreading Activation in a Network

Version 0.3.0

Description The notion of spreading activation is a prevalent metaphor in the cognitive sciences. This package provides the tools for cognitive scientists and psychologists to conduct computer simulations that implement spreading activation in a network representation. The algorithmic method implemented in 'spreadr' subroutines follows the approach described in Vitevitch, Ercal, and Adagarla (2011, Frontiers), who viewed activation as a fixed cognitive resource that could spread among nodes that were connected to each other via edges or connections (i.e., a network). See Vitevitch, M. S., Ercal, G., & Adagarla, B. (2011). Simulating retrieval from a highly clustered network: Implications for spoken word recognition. Frontiers in Psychology, 2, 369. <doi:10.3389/fpsyg.2011.00369> and Siew, C. S. Q. (2019). spreadr: A R package to simulate spreading activation in a network. Behavior Research Methods, 51, 910-929. <doi:10.3758/s13428-018-1186-5>.

License GPL-3

Encoding UTF-8

Depends Rcpp (>= 0.12.5), R (>= 2.10)

Imports Matrix, assertthat, igraph

Suggests dplyr, fs, ggplot2, gganimate, ggraph, gifski, knitr, rmarkdown, testthat (>= 3.0.0)

LazyData true

VignetteBuilder knitr

LinkingTo Rcpp

NeedsCompilation yes

Author Cynthia Siew [aut, cre], Dirk U. Wulff [ctb], Ning Yuan Lee [ctb]

Maintainer Cynthia Siew <cynsiewsq@gmail.com>

Config/testthat/edition 3

Config/roxygen2/version 8.0.0

URL https://csqsiew.github.io/spreadr/

Config/pak/sysreqs libglpk-dev libxml2-dev

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

Date/Publication 2026-05-27 07:04:50 UTC

RemoteUrl https://github.com/csqsiew/spreadr

RemoteRef HEAD

RemoteSha 602643325b7561e533f7547698ce4cdd6f800319

Contents

pnet 2
pnetm 3
spreadr 3

Index **6**

`pnet` *Small example of a phonological network as an igraph object*

Description

Small example of a phonological network as an igraph object

Usage

```
pnet
```

Format

igraph object representing an unweighted undirected graph with 34 vertices and 96 edges. There are no self-loops.

Source

Ying, Chan & Vitevitch, Michael. (2009). The Influence of the Phonological Neighborhood Clustering Coefficient on Spoken Word Recognition. Journal of experimental psychology. Human perception and performance. 35. 1934-49. 10.1037/a0016902.

pnetm *Small example of a phonological network as an adjacency matrix*

Description

Small example of a phonological network as an adjacency matrix

Usage

```
pnetm
```

Format

Adjacency matrix representing an unweighted undirected graph with 34 vertices and 96 edges. There are no self-loops.

Source

Ying, Chan & Vitevitch, Michael. (2009). The Influence of the Phonological Neighborhood Clustering Coefficient on Spoken Word Recognition. Journal of experimental psychology. Human perception and performance. 35. 1934-49. 10.1037/a0016902.

spreadr *Simulate spreading activation in a network*

Description

Simulate spreading activation in a network

Usage

```
spreadr(
  network,
  start_run,
  retention = 0.5,
  time = 10,
  threshold_to_stop = NULL,
  decay = 0,
  suppress = 0,
  include_t0 = FALSE,
  create_names = TRUE,
  never_stop = FALSE
)
```

Arguments

network	Adjacency matrix or igraph object representing the network in which to simulate spreading activation.
start_run	Non-empty data.frame with mandatory columns *node*, *activation*; and optional columns *time*. If the *time* column is present, *activation* is added to *node* at each *time*. Otherwise, the *activation*s are added to their corresponding nodes at t = 0.
retention	Number from 0 to 1 (inclusive) or a numeric vector of such numbers of length equals number of nodes in the network. This represents the proportion of activation that remains in the node (not spread) at each time step. Then, 1 - retention of the activation at each node is spread to neighbouring nodes. If a numeric vector, retentions are assigned to nodes according to the order given by V(network) if network is an igraph object or nrow(network) if network is an adjacency matrix.
time	Positive non-zero integer, or NULL. If not NULL, the number of time steps to simulate before stopping. Otherwise, stop with the threshold_to_stop parameter.
threshold_to_stop	Number or NULL. If not NULL, stop the simulation only when all nodes have activation value less than threshold_to_stop. Otherwise, stop with the time parameter.
decay	Number from 0 to 1 (inclusive) representing the proportion of activation that is lost at each time step.
suppress	Number representing the maximum amount of activation in a node for it to be set to 0, at each time step.
include_t0	Boolean flag indicating if activation at t = 0 should be prepended to the output data.frame. This is FALSE by default for back-compatibility.
create_names	Boolean flag indicating if nodes should be automatically named (1:n, where n is the number of nodes) in case they are missing.
never_stop	Boolean flag indicating if the simulation should be stopped if there have been too many iterations (so that there might be an infinite loop).

Details

At least one of parameters time or threshold_to_stop must be non-NULL. If both are non-NULL, the simulation stops at the earliest time possible.

The simulation iterates like so: for every i in [0, time],

- Spread activation from node to node
- Decay the activation at each node by the proportion specified by decay
- Set the activation at nodes with activation less than suppress to 0
- Add the activations in start_run with time = i to their corresponding nodes
- Save the activations at each node for output
- Check the terminating conditions time and threshold_to_stop. If any are satisfied, terminate the simulation.

Value

A data.frame with *node*, *activation* and *time* columns representing the spread of activation in the network over time.

Examples

```
# make an adjacency matrix and randomly fill some cells with 1s
mat <- matrix(sample(c(0,1), 100, replace=TRUE), 10, 10)
diag(mat) <- 0 # remove self-loops
initial_df <- data.frame(node=1, activation=20, stringsAsFactors=FALSE)
results <- spreadr(mat, initial_df)

head(results, 10)
tail(results, 10)
```

Index

∗ **datasets**
    pnet, 2
    pnetm, 3

data.frame, *4, 5*

igraph, *4*

pnet, 2
pnetm, 3

spreadr, 3