

Package: SINT (via r-universe)

October 9, 2026

Title Simulation and Analysis of Social Influence Network Models

Version 0.1.0

Description Tools for specifying, analyzing and simulating models of social influence network theory based on the Friedkin-Johnsen model, Friedkin and Johnsen (1990) [<doi:10.1080/0022250X.1990.9990069>](https://doi.org/10.1080/0022250X.1990.9990069), which includes the consensus model of DeGroot (1974) [<doi:10.1080/01621459.1974.10480137>](https://doi.org/10.1080/01621459.1974.10480137) as a special case. Equilibrium opinions, total influence matrices and convergence diagnostics are computed in closed form, also for signed networks with antagonistic ties, Altafini (2013) [<doi:10.1109/TAC.2012.2224251>](https://doi.org/10.1109/TAC.2012.2224251). Simulations allow influence weights and susceptibilities to depend on time and on the state of the system, and can couple latent opinions with manifest responses through logistic or threshold response functions, whose results are aggregated into collective outcomes by quota rules.

URL <https://github.com/Silvestro26/SINT>

BugReports <https://github.com/Silvestro26/SINT/issues>

License GPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

Suggests igraph, knitr, network, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

Config/roxygen2/version 8.1.0

VignetteBuilder knitr

Repository <https://silvestro26.r-universe.dev>

Date/Publication 2026-10-08 19:11:49 UTC

RemoteUrl <https://github.com/silvestro26/sint>

RemoteRef HEAD

RemoteSha a1b8e93cd23f1eac5fe6cdaab529609e3c7f617e

Contents

aggregate_quota	2
climate_balance	3
fj_check	3
fj_equilibrium	4
fj_influence	5
fj_simulate	6
influence_matrix	7
response_logistic	8
response_threshold	9
row_normalize	10
sint_simulate	10
Index	12

aggregate_quota	<i>Aggregate manifest responses with a quota rule</i>
-----------------	---

Description

Returns $\Theta(\sum_i 1(P_i = 1) - qn)$, where Θ is the Heaviside step function (1 for positive arguments, 0 otherwise). Only responses equal to 1 count in favor; abstentions (0) and opposing responses (-1) count as not in favor. With $quota = 0.5$ this is the simple majority rule.

Usage

```
aggregate_quota(P, quota = 0.5)
```

Arguments

P	Vector of manifest responses, or a matrix with one row per time step as returned by sint_simulate() .
quota	Quota q in $[0, 1]$.

Value

An integer (0 or 1), or an integer vector with one outcome per row when P is a matrix.

Examples

```
aggregate_quota(c(1, 1, 0, -1, 1))
aggregate_quota(c(1, 1, 0, -1, 1), quota = 2/3)
```

climate_balance	<i>Normative climate from manifest responses</i>
-----------------	--

Description

Computes $S = (n^+ - n^-)/(n^+ + n^- + \epsilon)$, where n^+ and n^- count the responses equal to +1 and -1.

Usage

```
climate_balance(P, eps)
```

Arguments

P	Vector of manifest responses in $\{-1, 0, +1\}$.
eps	Positive regularization constant ϵ .

Value

A single number in $(-1, 1)$.

Examples

```
climate_balance(c(1, 1, 1, 0, 0, 0, 0, 0, 0, 0), eps = 0.1)

# As a normative pressure in a response function
f <- response_threshold(delta = 0.5, theta = 0.4, gamma = 0.9,
  S = function(P) climate_balance(P, eps = 0.1))
```

fj_check	<i>Check the inputs of a Friedkin-Johnsen model</i>
----------	---

Description

Validates an influence matrix and a susceptibility vector and reports the quantities that govern convergence of the Friedkin-Johnsen dynamics $y(t+1) = \Lambda W y(t) + (I - \Lambda)y(0)$.

Usage

```
fj_check(W, lambda, tol = sqrt(.Machine$double.eps))
```

Arguments

<code>W</code>	Square numeric influence matrix.
<code>lambda</code>	Numeric vector of susceptibilities in $[0, 1]$, of length one (recycled) or equal to <code>nrow(W)</code> .
<code>tol</code>	Numerical tolerance used when checking row sums.

Details

A non-negative W must be row-stochastic. A signed W (at least one negative entry) must have absolute row sums not greater than one.

Value

A list with elements:

`n` Number of agents.

`signed` TRUE if W has negative entries.

`spectral_radius` Spectral radius of ΛW .

`norm_inf` Infinity norm of ΛW .

`converges` TRUE if the spectral radius is below one, so that the dynamics converge to a unique fixed point.

Examples

```
W <- matrix(c(0.5, 0.5, 0,
              0.2, 0.6, 0.2,
              0, 0.3, 0.7), nrow = 3, byrow = TRUE)
fj_check(W, lambda = c(0.8, 0.5, 0.9))
```

fj_equilibrium

Equilibrium opinions of a Friedkin-Johnsen model

Description

Computes the unique fixed point $y^* = (I - \Lambda W)^{-1}(I - \Lambda)y(0)$ of the dynamics $y(t + 1) = \Lambda W y(t) + (I - \Lambda)y(0)$. An error is raised when the spectral radius of ΛW is not below one.

Usage

```
fj_equilibrium(W, lambda, y0, tol = sqrt(.Machine$double.eps))
```

Arguments

W	Square numeric influence matrix.
lambda	Numeric vector of susceptibilities in $[0, 1]$, of length one (recycled) or equal to <code>nrow(W)</code> .
y0	Numeric vector of initial opinions, of length <code>nrow(W)</code> .
tol	Numerical tolerance used when checking row sums.

Value

A numeric vector of equilibrium opinions.

See Also

[fj_influence\(\)](#), [fj_simulate\(\)](#)

Examples

```
W <- matrix(c(0.5, 0.5, 0,
              0.2, 0.6, 0.2,
              0, 0.3, 0.7), nrow = 3, byrow = TRUE)
fj_equilibrium(W, lambda = c(0.8, 0.5, 0.9), y0 = c(0, 0.5, 1))
```

fj_influence

Total influence matrix of a Friedkin-Johnsen model

Description

Computes $V = (I - \Lambda W)^{-1}(I - \Lambda)$, whose entry v_{ij} is the total effect of the initial opinion of agent j on the equilibrium opinion of agent i . When W is non-negative, the rows of V sum to one.

Usage

```
fj_influence(W, lambda, tol = sqrt(.Machine$double.eps))
```

Arguments

W	Square numeric influence matrix.
lambda	Numeric vector of susceptibilities in $[0, 1]$, of length one (recycled) or equal to <code>nrow(W)</code> .
tol	Numerical tolerance used when checking row sums.

Value

A numeric $n \times n$ matrix.

See Also

[fj_equilibrium\(\)](#), [fj_check\(\)](#)

Examples

```
W <- matrix(c(0.5, 0.5, 0,
              0.2, 0.6, 0.2,
              0, 0.3, 0.7), nrow = 3, byrow = TRUE)
fj_influence(W, lambda = c(0.8, 0.5, 0.9))
```

fj_simulate

Simulate the Friedkin-Johnsen dynamics

Description

Iterates $y(t+1) = \Lambda W y(t) + (I - \Lambda)y(0)$ until the largest absolute change between two steps falls below `tol` or `max_steps` is reached. Unlike [fj_equilibrium\(\)](#), the iteration is carried out even when the dynamics do not converge.

Usage

```
fj_simulate(W, lambda, y0, max_steps = 1000L, tol = 1e-10)
```

Arguments

<code>W</code>	Square numeric influence matrix.
<code>lambda</code>	Numeric vector of susceptibilities in $[0, 1]$, of length one (recycled) or equal to <code>nrow(W)</code> .
<code>y0</code>	Numeric vector of initial opinions, of length <code>nrow(W)</code> .
<code>max_steps</code>	Maximum number of iterations.
<code>tol</code>	Convergence tolerance on the largest absolute change between two consecutive steps.

Value

A list with elements:

`trajectory` Matrix with one row per time step, from $t = 0$ to the last step, and one column per agent.

`steps` Number of iterations performed.

`converged` TRUE if the stopping criterion was met before `max_steps`.

See Also

[fj_equilibrium\(\)](#)

Examples

```
W <- matrix(c(0.5, 0.5, 0,
              0.2, 0.6, 0.2,
              0, 0.3, 0.7), nrow = 3, byrow = TRUE)
sim <- fj_simulate(W, lambda = c(0.8, 0.5, 0.9), y0 = c(0, 0.5, 1))
tail(sim$trajectory, 1)
```

influence_matrix	<i>Build an influence matrix from a network</i>
------------------	---

Description

Converts a matrix, an igraph graph or a network object into a row-normalized influence matrix suitable for the other functions of the package. The adjacency matrix, optionally valued with an edge attribute, is normalized with [row_normalize\(\)](#).

Usage

```
influence_matrix(x, weights = NULL, direction = c("attention", "influence"))
```

Arguments

x	A square numeric matrix, an igraph graph or a network object.
weights	Optional name of a numeric edge attribute holding the tie weights; if NULL, all ties have weight one. Ignored when x is a matrix.
direction	Meaning of an edge from <i>i</i> to <i>j</i> : "attention" (<i>i</i> attends to <i>j</i>) or "influence" (<i>i</i> influences <i>j</i>).

Details

With `direction = "attention"`, an edge from *i* to *j* means that *i* attends to *j*, so that $w_{ij} > 0$; this matches the row-wise reading of *W*. With `direction = "influence"`, an edge from *i* to *j* means that *i* influences *j*, and the adjacency matrix is transposed.

For network objects, self-ties are present only if the object was created with `loops = TRUE`.

Agents with no ties in their row cannot be normalized. They are given a unit self-weight, so that they attend only to themselves, and a warning is issued.

Value

A numeric matrix whose rows have unit absolute sums, with vertex names as dimnames when available.

See Also

[row_normalize\(\)](#), [fj_check\(\)](#)

Examples

```

A <- matrix(c(0, 1, 1,
              1, 0, 0,
              0, 1, 1), nrow = 3, byrow = TRUE)
influence_matrix(A)
influence_matrix(A, direction = "influence")

if (requireNamespace("igraph", quietly = TRUE)) {
  g <- igraph::graph_from_literal(a -- b, b -- c, c -- a, a -- c)
  influence_matrix(g)
}

```

response_logistic	<i>Logistic response function</i>
-------------------	-----------------------------------

Description

Builds a response function mapping latent opinions to binary manifest responses through $\Pr(P_i = 1) = \text{logit}^{-1}(\beta_i(y_i - \delta) + \gamma_i S_i)$.

Usage

```
response_logistic(beta = 1, delta = 0, gamma = 0, S = 0, stochastic = TRUE)
```

Arguments

beta	Discrimination β , of length one or n .
delta	Adhesion threshold δ .
gamma	Sensitivity γ to normative pressure, of length one or n .
S	Normative pressure: a number, a vector of length n , or a function of the previous manifest state.
stochastic	If TRUE, responses are drawn from Bernoulli distributions; if FALSE, the response probabilities are returned.

Details

S may be a fixed number (or vector) or a function of the previous manifest state P, such as [climate_balance\(\)](#). When S is a function and no previous manifest state exists (P is NULL), S is taken as 0.

Value

A function with arguments (t, y, P) for use in [sint_simulate\(\)](#).

See Also

[response_threshold\(\)](#), [sint_simulate\(\)](#)

Examples

```
f <- response_logistic(beta = 4, delta = 0.5, stochastic = FALSE)
f(1, y = c(0.2, 0.5, 0.9), P = NULL)
```

response_threshold	<i>Double-threshold response function</i>
--------------------	---

Description

Builds a response function mapping latent opinions to ternary manifest responses. With $z_i = \beta_i(y_i - \delta) + \gamma_i S_i$, the response is +1 if $z_i > \theta$, -1 if $z_i < -\theta$ and 0 (silence) otherwise.

Usage

```
response_threshold(
  beta = 1,
  delta = 0,
  theta,
  gamma = 0,
  S = 0,
  stochastic = FALSE
)
```

Arguments

beta	Discrimination β , of length one or n .
delta	Adhesion threshold δ .
theta	Positive half-width θ of the silence band.
gamma	Sensitivity γ to normative pressure, of length one or n .
S	Normative pressure: a number, a vector of length n , or a function of the previous manifest state.
stochastic	If TRUE, a logistic error is added to z_i .

Details

In the stochastic version a standard logistic error is added to z_i , which yields an ordered logit model with thresholds $\pm\theta$. S is handled as in [response_logistic\(\)](#).

Value

A function with arguments (t, y, P) for use in [sint_simulate\(\)](#).

See Also

[response_logistic\(\)](#), [climate_balance\(\)](#), [sint_simulate\(\)](#)

Examples

```
f <- response_threshold(delta = 0.5, theta = 0.4)
f(1, y = c(0.05, 0.5, 0.95), P = NULL)
```

row_normalize

Normalize the rows of an influence matrix

Description

Divides each row by the sum of the absolute values of its entries. For a non-negative matrix the result is row-stochastic; for a signed matrix the absolute row sums equal one, as required by [fj_check\(\)](#).

Usage

```
row_normalize(V)
```

Arguments

V Numeric matrix of unnormalized, possibly signed, weights.

Value

A numeric matrix with the same dimensions and names as *V*.

Examples

```
V <- matrix(c(2, 1, 1,
              0, 3, 1,
              1, 1, 2), nrow = 3, byrow = TRUE)
row_normalize(V)
```

sint_simulate

Simulate latent and manifest opinion dynamics

Description

Simulates a Friedkin-Johnsen process whose influence matrix and susceptibilities may depend on time and on the current state, coupled with an optional response function that maps latent opinions to manifest responses. At each step from t to $t + 1$:

$$y(t + 1) = \Lambda(t)W(t)y(t) + (I - \Lambda(t))y(0)$$

$$P(t + 1) = f(t + 1, y(t + 1), P(t))$$

where f is response.

Usage

```
sint_simulate(W, lambda, y0, steps, response = NULL, P0 = NULL)
```

Arguments

W	Influence matrix, or a function (t, y, P) returning one.
lambda	Susceptibility vector, or a function (t, y, P) returning one.
y0	Numeric vector of initial opinions, of length nrow(W).
steps	Number of steps to simulate.
response	Optional response function; if NULL, only the latent dynamics are simulated.
P0	Optional initial manifest state.

Details

W and lambda may be fixed objects or functions with arguments (t, y, P), called with the current time t , the latent state $y(t)$ and the manifest state $P(t)$ (NULL when there is no response function). They must return inputs valid for [fj_check\(\)](#).

response is a function with arguments (t, y, P), called with the new time $t+1$, the new latent state $y(t+1)$ and the previous manifest state $P(t)$; see [response_logistic\(\)](#) and [response_threshold\(\)](#). When P0 is NULL, the initial manifest state is computed as `response(0, y0, NULL)`.

External sources with fixed opinions can be represented as agents with susceptibility 0 and a unit self-weight.

Value

A list with elements:

y Matrix of latent opinions, one row per time step from $t = 0$ to steps, one column per agent.

P Matrix of manifest responses with the same layout, or NULL when response is NULL.

See Also

[fj_simulate\(\)](#), [response_threshold\(\)](#), [aggregate_quota\(\)](#)

Examples

```
W <- matrix(c(0.5, 0.5, 0,
             0.2, 0.6, 0.2,
             0, 0.3, 0.7), nrow = 3, byrow = TRUE)
f <- response_threshold(delta = 0.5, theta = 0.1, gamma = 0.2,
                      S = function(P) climate_balance(P, eps = 0.1))
sim <- sint_simulate(W, lambda = 0.8, y0 = c(0.1, 0.5, 0.9),
                   steps = 10, response = f)
sim$P
aggregate_quota(sim$P)
```

Index

aggregate_quota, [2](#)
aggregate_quota(), [11](#)

climate_balance, [3](#)
climate_balance(), [8](#), [9](#)

fj_check, [3](#)
fj_check(), [6](#), [7](#), [10](#), [11](#)
fj_equilibrium, [4](#)
fj_equilibrium(), [6](#)
fj_influence, [5](#)
fj_influence(), [5](#)
fj_simulate, [6](#)
fj_simulate(), [5](#), [11](#)

influence_matrix, [7](#)

response_logistic, [8](#)
response_logistic(), [9](#), [11](#)
response_threshold, [9](#)
response_threshold(), [8](#), [11](#)
row_normalize, [10](#)
row_normalize(), [7](#)

sint_simulate, [10](#)
sint_simulate(), [2](#), [8](#), [9](#)