Package ‘passt’

July 23, 2025

Type Package

Title Probability Associator Time (PASS-T)

Version 0.1.3

Description Simulates judgments of frequency and duration based on the Probability Associator Time (PASS-T) model. PASS-T is a memory model based on a simple competitive artificial neural network. It can imitate human judgments of frequency and duration, which have been extensively studied in cognitive psychology (e.g. Hintzman (1970) <doi:10.1037/h0028865>, Betsch et al. (2010) <`https://psycnet.apa.org/record/2010-18204-003`>). The PASS-T model is an extension of the PASS model (Sedlmeier, 2002, ISBN:0198508638). The package provides an easy way to run simulations, which can then be compared with empirical data in human judgments of frequency and duration.

License GPL-3

Encoding UTF-8

RoxygenNote 7.0.2

URL `https://github.com/johannes-titz/passt`

BugReports `https://github.com/johannes-titz/passt/issues`

Suggests knitr, ggplot2, plyr, testthat (>= 2.1.0), covr, markdown, rmarkdown

VignetteBuilder knitr

Imports magrittr,methods,dplyr,tidyr,rlang

NeedsCompilation no

Author Johannes Titz [aut, cre]

Maintainer Johannes Titz <`johannes.titz@gmail.com`>

Repository CRAN

Date/Publication 2021-05-03 14:30:02 UTC

Contents

run_exp 2
run_sim 3

Index **5**

run_exp *Run simulations and analyze data*

Description

Runs several simulations and returns correlative effect sizes between the frequency/total duration/single duration of each pattern and the output activation of the network for each pattern, respectively. Comparable to running an empirical experiment in judgments of frequency and duration and analyzing the data.

Usage

```
run_exp(
  frequency,
  duration,
  lrate_onset,
  lrate_drop_time,
  lrate_drop_perc,
  patterns = diag(length(duration)),
  number_of_participants = 100,
  cor_noise_sd = 0
)
```

Arguments

`frequency` presentation frequency for each pattern in the matrix

`duration` presentation duration for each pattern in the matrix

`lrate_onset` learning rate at the onset of a stimulus

`lrate_drop_time` point at which the learning rate drops, must be lower than duration

`lrate_drop_perc` how much the learning rate drops at lrate_drop_time

`patterns` matrix with input patterns, one row is one pattern

`number_of_participants` corresponds with number of simulations run

`cor_noise_sd` the amount of noise added to the final activations of the network, set to 0 if you do not want any noise

Value

data frame with three columns: f_dv, td_dv, t_dv which are the correlations between the frequency/total duration/single duration of each pattern and the activation of the network for each pattern, respectively.

See Also

run_sim

Examples

```
run_exp(10:1, 1:10, 0.05, 2, 0.2)
```

run_sim *Run simulations*

Description

Runs several simulations and returns output activation for each simulation and each input pattern

Usage

```
run_sim(
  patterns,
  frequency,
  duration,
  lrate_onset,
  lrate_drop_time,
  lrate_drop_perc,
  n_runs = 100,
  n_output_units = ncol(patterns),
  pulses_per_second = 1
)
```

Arguments

`patterns`	matrix with input patterns, one row is one pattern
`frequency`	presentation frequency for each pattern in the matrix
`duration`	presentation duration for each pattern in the matrix
`lrate_onset`	learning rate at the onset of a stimulus
`lrate_drop_time`	point at which the learning rate drops, must be lower than duration
`lrate_drop_perc`	how much the learning rate drops at lrate_drop_time
`n_runs`	number of simulations to be run, default is 100
`n_output_units`	number of output units, defaults to number of input units
`pulses_per_second`	how many time steps should be simulated per second

Value

list with following elements

- output: the sum of the activation strengths of the output units for each input pattern
- weight_matrix: final weight_matrix
- pres_matrix: presentation matrix

See Also

run_exp

Examples

```
run_sim(diag(10), 1:10, 10:1, 0.05, 2, 0.2)
```

Index

run_exp, *2*, *4*
run_sim, *3*, 3