Package ‘LATERmodel’

May 7, 2026

Title Linear Approach to Threshold with Ergodic Rate for Reaction Times

Version 0.2.0

Description Implements the Linear Approach to Threshold with Ergodic Rate (LATER) model, which predicts distributions of reaction times and summarises them with as little as two free parameters. Allows for easy visualisation and comparison of datasets, along with fitting of datasets using the LATER model.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.1

Imports dplyr, ggplot2, rlang, stats, tibble, parallel, moments

Depends R (>= 4.1)

LazyData true

Suggests lintr, styler, testthat (>= 3.0.0), vdiffr, printr, withr

Config/testthat/edition 3

URL `https://github.com/unimelbmdap/LATERmodel`, `https://unimelbmdap.github.io/LATERmodel/`

BugReports `https://github.com/unimelbmdap/LATERmodel/issues`

Config/Needs/website rmarkdown

NeedsCompilation no

Author Maria del Mar Quiroga [aut, cre] (ORCID: <`https://orcid.org/0000-0002-8943-2808`>),
Damien Mannion [aut] (ORCID: <`https://orcid.org/0000-0001-5520-632X`>),
Edoardo Tescari [aut] (ORCID: <`https://orcid.org/0000-0003-1157-4897`>),
Andrew Anderson [aut] (ORCID: <`https://orcid.org/0000-0001-7015-0061`>),
The University of Melbourne [cph, fnd]

Maintainer Maria del Mar Quiroga <`mar.quiroga@unimelb.edu.au`>

Repository CRAN

Date/Publication 2024-05-03 13:20:02 UTC

Contents

`carpenter_williams_1995` *Digitised data corresponding to Figure 2 of Carpenter and Williams (1995)*

Description

Digitised data corresponding to Figure 2 of Carpenter and Williams (1995)

Usage

```
carpenter_williams_1995
```

Format

`carpenter_williams_1995`**:**

A dataframe of 20014 reaction times for participant a and 22518 reaction times for participant b

participant Participant "name", either "a" or "b"

condition Prior percentage probability of the target being in the location of the eye movement (p05, p10, p25, p50, p75, p90, or p95)

time Saccadic latency in ms

Source

Carpenter, R. H., & Williams, M. L. L. (1995). Neural computation of log likelihood in control of saccadic eye movements. Nature, 377(6544), 59-62.

`compare_fits` *Compares the goodness-of-fit of a set of fit outcomes.*

Description

Compares the goodness-of-fit of a set of fit outcomes.

Usage

```
compare_fits(fits)
```

Arguments

`fits`	A list where each item has a name that identifies the fit and a value given by the output of `LATERmodel::fit_data`.

Details

The 'evidence ratio' is calculated as per Motulsky & Christopolous (2004), p. 146.

Value

A list of fit comparison results, ordered such that the fit with the lowest AIC value is in the first row.

- `aic` contains the fit AIC values.
- `preferred_rel_fit_delta_aic` is the AIC value for the fit relative to the AIC of the fit with the lowest AIC (preferred AIC - current AIC).
- `preferred_rel_fit_evidence_ratio` is the evidence ratio for the fit with the lowest AIC relative to the current fit.
- `preferred` is a boolean that indicates whether the fit has the lowest AIC value among the fits (is 'preferred').

Examples

```
data <- rbind(
  data.frame(name = "test", time = 1000/rnorm(100, 3, 1)),
  data.frame(name = "test_2", time = 1000/rnorm(100, 1, 1))
) |> dplyr::filter(time > 0)
data <- prepare_data(data)
fit_a <- fit_data(data = data, share_a = TRUE)
fit_b <- fit_data(data = data, share_sigma = TRUE)
comparison <- compare_fits(fits = list(a = fit_a, b = fit_b))
```

`fit_data` *Fit a LATER model to a single dataset or a pair of datasets.*

Description

Fit a LATER model to a single dataset or a pair of datasets.

Usage

```
fit_data(
  data,
  share_a = FALSE,
  share_sigma = FALSE,
  share_sigma_e = FALSE,
  with_early_component = FALSE,
  intercept_form = FALSE,
  use_minmax = FALSE,
  fit_criterion = "likelihood",
  jitter_settings = list(n = 7, prop = 0.5, seed = NA, processes = 2)
)
```

Arguments

`data`	A data frame with columns `name` and `promptness`.
`share_a`, `share_sigma`, `share_sigma_e`	If `FALSE` (the default), each dataset has its own parameter. If `TRUE`, the datasets share the relevant parameter.
`with_early_component`	If `TRUE`, the model contains a second 'early' component that is absent when `FALSE` (the default).
`intercept_form`	If `FALSE` (the default), the `a` parameter describes the mu parameter in the model; if `TRUE`, the `a` parameter describes the `k` parameter in the model (the intercept).
`use_minmax`	If `FALSE` (the default), the optimiser targets the sum of the goodness-of-fit values across datasets; if `TRUE`, it instead targets the maximum of the goodness-of-fit values across datasets.
`fit_criterion`	String indicating the criterion used to optimise the fit by seeking its minimum. • `ks`: Kolmogorov-Smirnov statistic. • `likelihood`: Negative log-likelihood.
`jitter_settings`	Settings for running the fitting multiple times with randomly-generated offsets ('jitter') applied to the starting estimates. • `n`: How many jitter iterations to run (default of 7); the total number of fits is `n + 1` (because the un-jittered start points are also fit). • `prop`: The maximum jitter offset, as a proportion of the start value (default of 0.5).

- `seed`: Seed for the random jitter generator (default is unseeded).
- `processes`: Maximum number of CPU processes that can be used (default is 2).

Value

A list of fitting arguments and outcomes.

- `fitted_params` is a named list of fitted parameter values.
- `named_fit_params` is a data frame with rows given by the dataset names and columns given by the parameter names.
- `loglike` is the overall log-likelihood of the fit.
- `aic` is the "Akaike's 'An Information Criterion'" value for the model.
- `optim_result` is the raw output from `stats::optim` for the best fit.
- `jitter_optim_results` contains the raw output from each call to `stats::optim` for the different start points.

Examples

```
data <- data.frame(name = "test", promptness = rnorm(100, 3, 1))
data_other <- data.frame(name = "test_2", promptness = rnorm(100, 1, 1))
fit_shared_sigma <- fit_data(
  data = rbind(data, data_other), share_sigma = TRUE
)
```

`individual_later_fit` *Fit individual LATER model to each dataset in a dataframe of datasets*

Description

Fit individual LATER model to each dataset in a dataframe of datasets

Usage

```
individual_later_fit(
  df,
  with_early_component = FALSE,
  fit_criterion = "likelihood",
  jitter_settings = list(n = 7, prop = 0.5, seed = NA, processes = 2)
)
```

Arguments

`df`	A dataframe with columns: `time`, `name`, `promptness`, and `e_cdf`
`with_early_component`	If `TRUE`, the model contains a second 'early' component that is absent when `FALSE` (the default).
`fit_criterion`	String indicating the criterion used to optimise the fit by seeking its minimum. • `ks`: Kolmogorov-Smirnov statistic. • `neg_loglike`: Negative log-likelihood.
`jitter_settings`	Settings for running the fitting multiple times with randomly-generated offsets ('jitter') applied to the starting estimates. • `n`: How many jitter iterations to run (default of 7). • `prop`: The maximum jitter offset, as a proportion of the start value (default of 0.5). • `seed`: Seed for the random jitter generator (default is unseeded). • `processes`: Maximum number of CPU processes that can be used (default is 2).

Value

A dataframe with one row for each named dataset in `df` and columns equal to the LATER model parameters returned by fit_data$named_fit_params

Examples

```
data <- rbind(
  data.frame(name = "test", promptness = rnorm(100, 3, 1)),
  data.frame(name = "test_2", promptness = rnorm(100, 1, 1))
)
fit_params <- individual_later_fit(data)
```

`ks_compare` *Apply two-sample KS test to all pairs of datasets contained within a dataframe*

Description

Apply two-sample KS test to all pairs of datasets contained within a dataframe

Usage

```
ks_compare(df, correct_multiple_comparisons = TRUE)
```

Arguments

`df`	A dataframe of datasets with columns: `name` and `time`, one unique `name` per dataset
`correct_multiple_comparisons`	If `TRUE`, an adjustment will be made to the p-values based on Holm, 1979, A simple sequentially rejective multiple test procedure

Value

A dataframe with columns `name1`, `name2`, `D`, and `p_value`

Examples

```
data <- prepare_data(dplyr::filter(
  carpenter_williams_1995,
  participant == "b"
))
ks_compare(data)
```

`ks_heatmap` *Create a heatmap to visualise if there is not enough evidence to reject the null hypothesis that two datasets come from the same underlying distribution*

Description

Create a heatmap to visualise if there is not enough evidence to reject the null hypothesis that two datasets come from the same underlying distribution

Usage

```
ks_heatmap(ks_results)
```

Arguments

`ks_results`	A dataframe with columns `name1`, `name2`, `D`, and `p-value`, obtained using the function ks_compare

Value

A heatmap plot with all paired comparisons

Examples

```
data <- prepare_data(dplyr::filter(
  carpenter_williams_1995,
  participant == "b"
))
ks_results <- ks_compare(data)
ks_heatmap(ks_results)
```

`model_cdf` *Evalulate the cumulative distribution function under the model.*

Description

Evalulate the cumulative distribution function under the model.

Usage

```
model_cdf(q, later_mu, later_sd, early_sd = NULL)
```

Arguments

`q`	Vector of quantiles
`later_mu`	Vector of the mean of the later component.
`later_sd`	Vector of the standard deviation of the later component.
`early_sd`	Vector of the standard deviation of the early component, or `NULL` if there is no early component (the default).

Value

Vector of cumulative distribution values

Examples

```
model_cdf(q = 1, later_mu = 1, later_sd = 1)
model_cdf(q = 1, later_mu = 1, later_sd = 1, early_sd = 3)
```

`model_pdf` *Evalulate the probability density function under the model.*

Description

Evalulate the probability density function under the model.

Usage

```
model_pdf(x, later_mu, later_sd, early_sd = NULL, log = FALSE)
```

Arguments

`x`	Vector of quantiles
`later_mu`	Vector of the mean of the later component.
`later_sd`	Vector of the standard deviation of the later component.
`early_sd`	Vector of the standard deviation of the early component, or `NULL` if there is no early component (the default).
`log`	If `TRUE`, probabilities are given as `log(p)`.

Value

Vector of probabilities

Examples

```
model_pdf(x = 1, later_mu = 1, later_sd = 1)
model_pdf(x = 1, later_mu = 1, later_sd = 1, early_sd = 3)
```

`prepare_data` *Prepares data for reciprobit plot*

Description

Prepares data for reciprobit plot

Usage

```
prepare_data(raw_data, time_unit = "ms", name_separator = "_")
```

Arguments

`raw_data`	Vector of reaction times for a single participant, or a dataframe containing a column called `time` with the reaction times and optional other columns: • a column called `name` with a unique label for each dataset • a column called `participant` with a unique id per participant and another called `condition` with a unique label for each condition. In this case the `name` for each dataset will be constructed as `participant+name_separator+condition`. • a column called `color` that contains one hexadecimal color code for each dataset. In this case the `name` for each dataset will be set to be the name of the color.
`time_unit`	Units of the reaction times in rt_vector, must be one of "ms", "ds", or "s".
`name_separator`	If the `raw_data` dataframe does not contain a `name` column, but does have `participant` and `condition` columns, the `name` for each dataset will be constructed as `participant+name_separator+condition`.

Value

A dataframe with columns: `time`, `color`, `name`, `promptness`, and `e_cdf`.

Examples

```
df <- prepare_data(carpenter_williams_1995)
```

`promptness_ecdf` *Compute the empirical cumulative distribution function for promptness*

Description

Compute the empirical cumulative distribution function for promptness

Usage

```
promptness_ecdf(promptness, adjust_for_times = TRUE, eval_unique = FALSE)
```

Arguments

`promptness`	A vector of promptness values (1 / times)
`adjust_for_times`	If `TRUE` (the default), the returned `y` value is such that `1 - y = P(1/promptness <= 1/x). If` FALSE`, th` = P(promptness <= x)‘.
`eval_unique`	If `FALSE` (the default), the ECDF is evaluated at all values in `promptness`. If `TRUE`, the ECDF is evaluated at the unique values in `promptness`.

Value

A data frame with attributes:

- `x` is the values at which the ECDF was evaluated.
- `y` is the evaluated value of the ECDF.

Examples

```
p <- promptness_ecdf(promptness = rnorm(100, 3, 1))
```

`reciprobit_plot` *Plot reaction times and LATER model fit in reciprobit axes*

Description

Plot reaction times and LATER model fit in reciprobit axes

Usage

```
reciprobit_plot(
  plot_data,
  fit_params = NULL,
  time_breaks = c(0.1, 0.2, 0.3, 0.5, 1),
  probit_breaks = c(0.1, 1, 5, 10, 20, 50, 80, 90, 95, 99, 99.9),
  z_breaks = c(-2, -1, 0, 1, 2),
  xrange = NULL,
  yrange = NULL
)
```

Arguments

`plot_data`	A dataframe with columns: `time`, `name`, `promptness`, and `e_cdf`. Optionally, there may be a `color` column, which contains hex values, one unique hex value per named dataset
`fit_params`	A dataframe with one row for each named dataset and columns equal to the LATER model parameters returned by `fit_data$named_fit_params`
`time_breaks`	Desired tick marks on the x axis, expressed in promptness (1/s)
`probit_breaks`	Desired tick marks on the y axis in probit space
`z_breaks`	Desired tick marks on secondary y axis, in z values
`xrange`	Desired range for the x axis, in promptness (1/s)
`yrange`	Desired range for the y axis, in cumulative probability space

Value

A reciprobit plot with the cumulative probability distribution of the reaction times

Examples

```
data <- rbind(
  data.frame(name = "test", time = 1000/rnorm(100, 3, 1)),
  data.frame(name = "test_2", time = 1000/rnorm(100, 4, 1))
) |> dplyr::filter(time > 0)
data <- prepare_data(data)
fit_params <- individual_later_fit(data)
reciprobit_plot(data, fit_params)
```

`reddi_asrress_carpenter_2003` *Digitised data corresponding to Figure 2 of Reddi, Asrress, and Carpenter, 2003.*

Description

Digitised data corresponding to Figure 2 of Reddi, Asrress, and Carpenter, 2003.

Usage

```
reddi_asrress_carpenter_2003
```

Format

`reddi_asrress_carpenter_2003`:

A dataframe of 1600 reaction times for participant J

participant Participant "name", equal to "J"

condition Different degrees of coherence of the kinematograms shown as stimuli, (64%, 32%, 16%, 8%: S, low-urgency–slow and accurate–conditions)

time Saccadic latency in ms

Source

Reddi, B. A., Asrress, K. N., & Carpenter, R. H. (2003). Accuracy, information, and response time in a saccadic decision task. Journal of Neurophysiology, 90(5), 3538-3546.

`simulate_dataset` *Simulate a dataset given model parameters.*

Description

Generates samples from a set of provided LATER model parameters, with the option to iteratively replace invalid samples (reaction times <= 0).

Usage

```
simulate_dataset(
  n,
  later_mu,
  later_sd,
  early_sd = NULL,
  seed = NA,
  allow_negative_times = FALSE
)
```

Arguments

`n`	Number of samples (trials)
`later_mu`	Mean of the later component.
`later_sd`	Standard deviation of the later component.

`early_sd`	Standard deviation of the early component, or `NULL` if there is no early component (the default).
`seed`	Seed for the random number generator
`allow_negative_times`	If `FALSE` (the default), any random samples that have negative response times are iteratively replaced such that all returned samples are positive. If `TRUE`, no such replacement is performed.

Value

Vector of response times (in seconds)

Examples

```
simulate_dataset(n = 100, later_mu = 5, later_sd = 1)
simulate_dataset(n = 100, later_mu = 5, later_sd = 1, early_sd = 5)
```

Index