

Package: rsprite2 (via r-universe)

October 10, 2026

Title Identify Distributions that Match Reported Sample Parameters (SPRITE)

Version 0.2.1.9002

Description The SPRITE algorithm creates possible distributions of discrete responses based on reported sample parameters, such as mean, standard deviation and range (Heathers et al., 2018, <[doi:10.7287/peerj.preprints.26968v1](https://doi.org/10.7287/peerj.preprints.26968v1)>). This package implements it, drawing heavily on the code for Nick Brown's 'rSPRITE' Shiny app <<https://shiny.ieis.tue.nl/sprite/>>. In addition, it supports the modeling of distributions based on multi-item (Likert-type) scales and the use of restrictions on the frequency of particular responses.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Suggests ggplot2, testthat (>= 3.0.0), tibble, tidyr, rlang, scales

Config/testthat/edition 3

URL <https://lukaswallrich.github.io/rsprite2/>

BugReports <https://github.com/LukasWallrich/rsprite2/issues>

Imports checkmate, Rdpack

RdMacros Rdpack

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

Date/Publication 2026-09-09 22:29:59 UTC

RemoteUrl <https://github.com/lukaswallrich/rsprite2>

RemoteRef HEAD

RemoteSha 78231f75751eb5f34cd53f5d8391a4baf7b2cb3b

Contents

boundary_test	2
find_possible_distribution	3
find_possible_distributions	4
GRIM_test	6
GRIMMER_test	7
plot_distributions	9
set_parameters	10
Index	13

boundary_test	<i>Boundary test for standard deviation</i>
---------------	---

Description

This function tests whether a given standard deviation is within the range that is theoretically possible for a given sample size and mean, assuming responses are bounded by a minimum and maximum scale value. It provides a simple TRUE/FALSE check. Note that SDs within this range may still be impossible due to their granularity, so for samples where the sample size (x number of items) is below 10 to the power of the precision of the reported SD, this should only be a precursor to GRIMMER_test()

Usage

```
boundary_test(  
  sd,  
  n_obs,  
  mean,  
  min_val,  
  max_val,  
  m_prec = NULL,  
  sd_prec = NULL,  
  n_items = 1,  
  return_range = FALSE,  
  quiet = FALSE  
)
```

Arguments

sd	The standard deviation of the distribution
n_obs	The number of observations (sample size), an integer of at least two for reconstruction.
mean	The mean of the distribution
min_val	The minimum value
max_val	The maximum value

m_prec	The precision of the reported mean, as a nonnegative integer number of decimal places (at most 308), consistent with the supplied value. Inferred from the numeric value if omitted; specify it explicitly when the reported mean ends in zero.
sd_prec	The precision of the reported standard deviation, with the same requirements as m_prec. Specify it explicitly to preserve trailing zeroes.
n_items	Positive integer number of items in scale, if distribution represents scale averages. Defaults to 1, which represents any single-item measure.
return_range	(Optional) If TRUE, the function returns a numeric vector with the SD possible range.
quiet	Suppress warnings.

Value

Logical TRUE if the standard deviation is within the possible range, and FALSE otherwise, unless return_range, in which case a numeric vector with the lower and upper bounds of the possible SD range is returned. If the range is undefined, return_range = TRUE returns two numeric NA values. Scale endpoints must be integers; multi-item averages have spacing $1 / n_items$.

Examples

```
# For a 7-point scale with N = 20 and Mean = 4, an SD of 1.5 is within the possible range
boundary_test(sd = 1.5, n_obs = 20, mean = 4, min_val = 1, max_val = 7)
# but an SD of 3.5 is not possible
boundary_test(sd = 3.5, n_obs = 20, mean = 4, min_val = 1, max_val = 7)
```

```
find_possible_distribution
```

Find a possible distribution.

Description

This function aims to find a possible distribution that would give rise to the observed sample parameters. For that, you need to pass a list of parameters, best created with [set_parameters](#)

Usage

```
find_possible_distribution(parameters, seed = NULL, values_only = FALSE)
```

Arguments

parameters	List of parameters, see set_parameters
seed	An integer to use as the seed for random number generation. Set this in scripts to ensure reproducibility. Explicit seeds leave the caller's random number generator state unchanged.
values_only	Should only values or a more informative list be returned. See Value section.

Details

A successful search supplies one compatible reconstruction. Search failure does not prove impossibility, and the frequencies with which reconstructions are found do not estimate probabilities for the original data. Rounding ties are accepted in either direction.

Value

Unless `values_only = TRUE`, a list with:

<code>outcome</code>	success or failure - character
<code>values</code>	The distribution that was found (if success) / whose SD came closest to the target during the search (if failure) - numeric
<code>mean</code>	The exact mean of the distribution - numeric
<code>sd</code>	The SD of the distribution that was found (success) / that came closest (failure) - numeric
<code>iterations</code>	The number of SD adjustments attempted; zero if the initial candidate succeeds. On failure, this is the total attempted, even if the closest candidate was found earlier.

If `values_only = TRUE`, then the distribution is returned if one was found, and `NULL` if it failed.

Examples

```
sprite_parameters <- set_parameters(mean = 2.2, sd = 1.3, n_obs = 20,
                                     min_val = 1, max_val = 5)
find_possible_distribution(sprite_parameters)
```

```
find_possible_distributions
```

Find several possible distributions.

Description

This function aims to find several possible distribution that would give rise to the observed sample parameters. For that, you need to pass a list of parameters, created with [set_parameters](#)

Usage

```
find_possible_distributions(
  parameters,
  n_distributions = 10,
  seed = NULL,
  return_tibble = TRUE,
  return_failures = FALSE
)
```

Arguments

parameters	List of parameters, see set_parameters
n_distributions	The target number of distributions to return.
seed	An integer to use as the seed for random number generation. Set this in scripts to ensure reproducibility. Explicit seeds leave the caller's random number generator state unchanged.
return_tibble	Should a tibble, rather than a list, be returned? Requires the tibble-package, ignored if that package is not available.
return_failures	Should distributions that failed to produce the desired SD be returned? Defaults to false

Details

A successful search supplies one compatible reconstruction. Search failure does not prove impossibility, and the frequencies with which reconstructions are found do not estimate probabilities for the original data. Rounding ties are accepted in either direction.

Value

A tibble or list (depending on the return_tibble argument) with:

outcome	success or failure - character
distribution	The distribution that was found (if success) / whose SD came closest to the target during the search (if failure) - numeric
mean	The exact mean of the distribution - numeric
sd	The SD of the distribution that was found (success) / that came closest (failure) - numeric
iterations	The number of SD adjustments attempted on the first search that found this distribution. Zero if its initial candidate succeeded; on failure, the total number attempted.

Examples

```
sprite_parameters <- set_parameters(mean = 2.2, sd = 1.3, n_obs = 20,  
                                     min_val = 1, max_val = 5)  
  
find_possible_distributions(sprite_parameters, 5, seed = 1234)
```

GRIM_test

*GRIM test for mean***Description**

This function tests whether a given mean (with a specific precision) can result from a sample of a given size based on integer responses to one or more items. The test is based on Brown & Heathers (2017).

Usage

```
GRIM_test(
  mean,
  n_obs,
  m_prec = NULL,
  n_items = 1,
  return_values = FALSE,
  return_list = FALSE,
  quiet = FALSE
)
```

Arguments

mean	The mean of the distribution
n_obs	The number of observations (sample size), an integer of at least two for reconstruction.
m_prec	The precision of the reported mean, as a nonnegative integer number of decimal places (at most 308), consistent with the supplied value. Inferred from the numeric value if omitted; specify it explicitly when the reported mean ends in zero.
n_items	Positive integer number of items in scale, if distribution represents scale averages. Defaults to 1, which represents any single-item measure.
return_values	A logical value. <i>ignored if</i> return_list = TRUE. If FALSE (the default), the function returns a simple TRUE or FALSE. If TRUE, it returns a numeric vector of all possible means that are consistent with the parameters. If the test fails when return_values = TRUE, a warning is issued and the closest consistent mean is returned. If there are two equally close means, both are returned.
return_list	A logical value. If FALSE (the default), the function's return type is determined by return_values. If TRUE, the function instead returns a list containing two elements: a logical passed flag and a numeric vector values containing the relevant means.
quiet	Suppress warnings.

Details

Logical-only requests compare interval endpoints without enumerating possible means. Value and list requests stop if more than one million candidates would need enumeration. Inputs beyond reliable integer arithmetic also stop with an informative error.

Value

The return type depends on the arguments. By default, a logical scalar (TRUE or FALSE). If `return_values = TRUE`, a numeric vector is returned. If `return_list = TRUE`, a list is returned.

References

Brown NJ, Heathers JA (2017). “The GRIM test: A simple technique detects numerous anomalies in the reporting of results in psychology.” *Social Psychological and Personality Science*, **8**(4), 363–369.

Examples

```
# A sample of 28 integers cannot result in a mean of 5.19.
GRIM_test(mean = 5.19, n_obs = 28)

# To find the closest possible mean, set return_values to TRUE
GRIM_test(mean = 5.19, n_obs = 28, return_values = TRUE)

# To get a detailed list containing both the logical result and the values
GRIM_test(mean = 5.19, n_obs = 28, return_list = TRUE)
```

GRIMMER_test

GRIMMER test for standard deviation

Description

This function tests whether a given standard deviation (with a specific precision) can result from a sample of a given size based on integer responses to one or more items. The test was first proposed by [Anaya \(2016\)](#); here, the algorithm developed by [Allard \(2018\)](#) is used, extended by Aurélien Allard to support multi-item scales.

Usage

```
GRIMMER_test(
  mean,
  sd,
  n_obs,
  m_prec = NULL,
  sd_prec = NULL,
  n_items = 1,
  min_val = NULL,
```

```

    max_val = NULL,
    return_values = FALSE,
    return_list = FALSE,
    quiet = FALSE
  )

```

Arguments

<code>mean</code>	The mean of the distribution
<code>sd</code>	The standard deviation of the distribution
<code>n_obs</code>	The number of observations (sample size), an integer of at least two for reconstruction.
<code>m_prec</code>	The precision of the reported mean, as a nonnegative integer number of decimal places (at most 308), consistent with the supplied value. Inferred from the numeric value if omitted; specify it explicitly when the reported mean ends in zero.
<code>sd_prec</code>	The precision of the reported standard deviation, with the same requirements as <code>m_prec</code> . Specify it explicitly to preserve trailing zeroes.
<code>n_items</code>	Positive integer number of items in scale, if distribution represents scale averages. Defaults to 1, which represents any single-item measure.
<code>min_val</code>	(Optional) Scale minimum. If provided alongside <code>max_val</code> , the function checks whether the SD is consistent with that range.
<code>max_val</code>	(Optional) Scale maximum.
<code>return_values</code>	A logical value, ignored if <code>return_list</code> = TRUE. If TRUE, return the unrounded SDs within the tested rounding interval that satisfy the GRIMMER arithmetic conditions. Otherwise return a logical verdict.
<code>return_list</code>	If TRUE, return a list with logical passed and numeric values components, irrespective of <code>return_values</code> .
<code>quiet</code>	Suppress warnings.

Details

GRIMMER compatibility is a necessary condition for a sample to exist, not proof that a sample exists. The integer, parity, and minimum-variance conditions on sums of squares can pass even when no sample produces the reported statistics. Optional scale bounds add a range check but do not make the test sufficient. For example, two integer observations on a scale from 0 to 4 cannot have mean 2 and SD 2.0, although these statistics pass GRIMMER.

Both endpoints of each reporting interval are accepted to accommodate different conventions for rounding ties. A reported SD of zero therefore also includes small positive SDs that round to zero.

Value

A logical scalar by default. With `return_values` = TRUE, a numeric vector of compatible unrounded SDs; an empty vector always means failure. With `return_list` = TRUE, a list with passed and values components. Candidate values satisfy the screening conditions and need not correspond to actual samples. Requests requiring more than one million candidates to be enumerated, or inputs beyond reliable integer arithmetic, stop with an informative error.

References

Anaya J (2016). “The GRIMMER test: A method for testing the validity of reported measures of variability.” *PeerJ Preprints*, 4, e2400v1. Allard A (2018). “Analytic-GRIMMER: A New Way of Testing the Possibility of Standard Deviations.” <https://aurelienallard.netlify.app/post/anaytic-grimmer-possibility-standard-deviations/>. Accessed: 2025-07-09.

Examples

```
# A sample of 18 integers with mean 3.44 cannot have an SD of 2.47.
GRIMMER_test(mean = 3.44, sd = 2.47, n_obs = 18)

# Get a detailed list containing both the logical result and any possible values
GRIMMER_test(mean = 3.5, sd = 1.21, n_obs = 10, return_list = TRUE)
```

plot_distributions	<i>Plot distributions</i>
--------------------	---------------------------

Description

This plots distributions identified by `find_possible_distributions` using ggplot2. They can be shown as histograms or as **cumulative distributions (ECDF) plots**. The latter give more information, yet not all audiences are familiar with them.

Usage

```
plot_distributions(
  distributions,
  plot_type = c("auto", "histogram", "ecdf", "density"),
  max_plots = 100,
  show_ids = FALSE,
  facets = NULL
)
```

Arguments

distributions	Nonempty tibble with a list-column distribution containing nonempty, finite numeric vectors, and an identifier (id), typically as returned from <code>find_possible_distributions</code> . Density plots require at least two distinct responses in each distribution.
plot_type	Plot multiple histograms, or overlapping cumulative distribution plots, or density plots? "auto" is to plot histograms if up to 9 distributions are passed, or if there are fewer than 10 discrete values, and empirical cumulative distribution plots otherwise
max_plots	Positive integer: how many distributions should <i>at most</i> be plotted? If more are passed, this number is randomly selected.
show_ids	Should ids of the distributions be shown with ecdf and density charts? Defaults to no, since the default ids are not meaningful.

facets Should distributions be shown in one chart or in multiple small charts? Only considered for ecdf and density charts, histograms are always shown in facets

Details

Histograms show the exact frequency at each observed response. Bar widths use the response spacing ($1 / n_items$) for SPRITE results. For ordinary tibbles, widths use the smallest observed spacing, or 1 if all responses are identical. Scale endpoints and complete bars are retained. Empty inputs produce an informative error.

Value

A ggplot2 object that can be styled with functions such as [labs](#) or [theme_linedraw](#)

Examples

```
sprite_parameters <- set_parameters(mean = 2.2, sd = 1.3, n_obs = 20,
                                   min_val = 1, max_val = 5)

poss <- find_possible_distributions(sprite_parameters, 5, seed = 1234)

# All distributions in same plot
plot_distributions(poss, plot_type = "ecdf")

# Separate plot for each distribution
plot_distributions(poss, plot_type = "ecdf", facets = TRUE)
```

set_parameters	<i>Define parameters for SPRITE algorithm</i>
----------------	---

Description

The SPRITE algorithm aims to construct possible distributions that conform to observed/reported parameters. This function performs some checks and returns a list of these parameters that can then be passed to the functions that actually generate the distributions (e.g. [find_possible_distribution](#))

Usage

```
set_parameters(
  mean,
  sd,
  n_obs,
  min_val,
  max_val,
  m_prec = NULL,
  sd_prec = NULL,
  n_items = 1,
```

```

    restrictions_exact = NULL,
    restrictions_minimum = NULL,
    dont_test = FALSE
)

```

Arguments

mean	The mean of the distribution
sd	The standard deviation of the distribution
n_obs	The number of observations (sample size), an integer of at least two for reconstruction.
min_val	The minimum value
max_val	The maximum value
m_prec	The precision of the reported mean, as a nonnegative integer number of decimal places (at most 308), consistent with the supplied value. Inferred from the numeric value if omitted; specify it explicitly when the reported mean ends in zero.
sd_prec	The precision of the reported standard deviation, with the same requirements as m_prec. Specify it explicitly to preserve trailing zeroes.
n_items	Positive integer number of items in scale, if distribution represents scale averages. Defaults to 1, which represents any single-item measure.
restrictions_exact	Restrictions on the exact frequency of specific responses, see Details
restrictions_minimum	Restrictions on the minimum frequency of specific responses, see Details
dont_test	By default, this function tests whether the mean is possible, given the sample size (GRIM-test) and whether the standard deviation is possible, given mean and sample size (GRIMMER test), and fails otherwise. If you want to override this, and run SPRITE anyway, you can set this to TRUE. This skips GRIM, GRIMMER, and SD-bound checks; input validation and restriction checks still apply.

Details

Restrictions can be used to define how often a specific value should appear in the sample. They need to be passed as a list in the form `value = frequency`. Thus, to specify that there should be no 3s and five 4s in the distribution, you would pass `restrictions_exact = list("3" = 0, "4" = 5)`. To specify that there should be at least one 1 and one 7, you would pass `restrictions_minimum = list("1" = 1, "7" = 1)`. If you just want to specify that the minimum and maximum values appear at least once (for instance when they are the reported rather than possible range), you can use the shortcut `restrictions_minimum = "range"`. Finally, for multi-item scales, name the response as precisely as needed to identify it. Exact lattice matches take priority; rounded names are also accepted if they identify a unique response (e.g., `list("1.67" = 0)` when `n_items = 3`). Ambiguous shorthand and duplicate restrictions are rejected. Counts must be nonnegative integers, with their sum no greater than `n_obs`.

Value

A named list of parameters, pre-processed for further rsprite2 functions.

Examples

```
set.seed(1234) #To get reproducible results

# Simple case
sprite_parameters <- set_parameters(mean = 2.2, sd = 1.3, n_obs = 20, min_val = 1, max_val = 5)
find_possible_distribution(sprite_parameters)

# With restrictions
sprite_parameters <- set_parameters(mean = 1.95, sd = 1.55, n_obs = 20,
                                   min_val = 1, max_val = 5, n_items = 3,
                                   restrictions_exact = list("3"=0, "3.67" = 2),
                                   restrictions_minimum = "range")
find_possible_distribution(sprite_parameters)
```

Index

`boundary_test`, [2](#)

`find_possible_distribution`, [3](#), [10](#)

`find_possible_distributions`, [4](#), [9](#)

`GRIM_test`, [6](#)

`GRIMMER_test`, [7](#)

`labs`, [10](#)

`plot_distributions`, [9](#)

`set_parameters`, [3–5](#), [10](#)

`theme_linedraw`, [10](#)