Package ‘gridsampler’

October 13, 2022

License GPL-3

Title A Simulation Tool to Determine the Required Sample Size for
Repertory Grid Studies

Type Package

LazyLoad yes

Description Simulation tool to facilitate determination of
required sample size to achieve category saturation
for studies using multiple repertory grids in conjunction with
content analysis.

Version 0.6

Date 2016-11-23

Imports shiny, ggplot2, reshape2, plyr, shinythemes, BiasedUrn,
shinyBS

Suggests knitr, testthat, rmarkdown

Encoding UTF-8

URL `https://github.com/markheckmann/gridsampler`

BugReports `https://github.com/markheckmann/gridsampler/issues`

VignetteBuilder knitr

RoxygenNote 5.0.1

NeedsCompilation no

Author Mark Heckmann [aut, cre],
Lukas Burk [aut]

Maintainer Mark Heckmann <`heckmann.mark@gmail.com`>

Repository CRAN

Date/Publication 2016-11-23 17:24:13

R topics documented:

`gridsampler-package` **gridsampler** - *A sample size simulation software for repertory grid studies*

Description

gridsampler - A sample size simulation software for repertory grid studies

References

- Green, B. (2004). Personal construct psychology and content analysis. Personal Construct Theory & Practice, 1(3), 82-91.
- Jankowicz, D. (2004). The easy guide to repertory grids. Chichester, England: John Wiley & Sons.

`calc_probabilities` *Probability for certain degree of saturation*

Description

Calculate probability for getting certain proportion of categories with at least m constructs

Usage

```
calc_probabilities(r, n, ms, min.props = c(0.9, 0.95, 0.99))
```

Arguments

`r`	A dataframe. The result returned from `sim_n_persons_x_times_many_n`.
`n`	Vector of n for which to calculate probabilities.
`ms`	minimal number of constructs in each category
`min.props`	Proportion of categores to contain at least m constructs.

See Also

Other Utilities: `expected_frequencies`, `prob_categories`

Examples

```
prob <-  dexp(1:30, .05)
n <- seq(10, 80, by = 20)
r <- sim_n_persons_x_times_many_n(prob, n, a = 7, times = 100)
dd <- calc_probabilities(r, n, ms=1:5, min.props = c(0.9, .95, 1))
head(dd)
```

`draw_multiple_n_persons_x_times` *Draw and redraw results of simulation*

Description

Draw and redraw results of simulation

Usage

```
draw_multiple_n_persons_x_times(d)
```

Arguments

`d`	A dataframe as returned by `calc_probabilities`.

See Also

Other Plotting: `draw_n_person_sample`

Examples

```
## simulate
prob <-  dexp(1:30, .05)      # probabilities for categories
N <- seq(10, 80, by = 10)     # smaple sizes to simulate
r <- sim_n_persons_x_times_many_n(prob, n = N, a = 7, times = 100, progress = "none")

# calculate and draw
M <- 1:5                      # minimal number of categories to evaluate
p <- c(0.9, .95, 1)           # proportion of categories for which minimal m holds
d <- calc_probabilities(r, n = N, ms = M, min.props = p)
draw_multiple_n_persons_x_times(d)
```

draw_n_person_sample *Produce graphic for a single sample of n persons*

Description

Produce graphic for a single sample of n persons

Usage

```
draw_n_person_sample(prob, n, a = 10, ap = rep(1/length(a), length(a)))
```

Arguments

prob	Probability to draw a construct from a certain category.
n	Number of persons, i.e. grids to be sampled.
a	Possible number of attributes sampled from.
ap	Attribute probabilities, i.e. for each number of attributes given in a.

See Also

Other Plotting: draw_multiple_n_persons_x_times

Examples

```
draw_n_person_sample(dexp(1:30, rate = .05), n = 100, a = 10)
draw_n_person_sample(dexp(1:30, rate = .05), n = 100, a = 1:5, ap = 5:1)
```

expected_frequencies *Produce ggplot of percentiles for simulated frequencies*

Description

Produce ggplot of percentiles for simulated frequencies

Usage

```
expected_frequencies(r)
```

Arguments

r	A dataframe. The result returned from sim_n_persons_x_times.

Value

Draws a ggplot

See Also

Other Utilities: `calc_probabilities`, `prob_categories`

Examples

```
r <- sim_n_persons_x_times(dexp(1:30, rate = .05), n = 50, a = 5:7, ap = 1:3, 100)
expected_frequencies(r)
```

`gridsampler` *Run gridsampler app*

Description

This function starts the gridsampler shiny app.

Usage

```
gridsampler(display.mode = "auto",
  launch.browser = getOption("shiny.launch.browser", interactive()))
```

Arguments

`display.mode`	`auto` by default, can also be `showcase`. See runApp.
`launch.browser`	Boolean, set `TRUE` to open the app in the browser. See runApp.

Examples

```
## Not run:
gridsampler()

## End(Not run)
```

`prob_categories` *Probability for certain degree of saturation*

Description

Calculate probability for getting certain proportion of categories with at least m constructs

Usage

```
prob_categories(r, m, min.prop = 1)
```

Arguments

`r` A dataframe. The result returned from `sim_n_persons_x_times`.

`m` minimal number of constructs in each category

`min.prop` Proportion of categores to contain at least m constructs.

See Also

Other Utilities: `calc_probabilities`, `expected_frequencies`

Examples

```
r <- sim_n_persons_x_times(dexp(1:30, rate = .05), n = 50, a = 5:7, times = 100, progress = "none")
prob_categories(r, 4, min.prop = .9)
```

`sim_n_persons` *Simulate n persons*

Description

Function is a simple replicate wrapper around `sim_one_person`

Usage

```
sim_n_persons(prob, n, a = 10, ap = rep(1/length(a), length(a)))
```

Arguments

`prob` Probability to draw a construct from a certain category.

`n` Number of persons, i.e. grids to be sampled.

`a` Possible number of attributes sampled from.

`ap` Attribute probabilities, i.e. for each number of attributes given in `a`.

See Also

Other Simulations: `sim_n_persons_x_times_many_n`, `sim_n_persons_x_times`, `sim_one_person`

Examples

```
sim_n_persons(dexp(1:30, .05), n = 2, a = 10)
sim_n_persons(dexp(1:30, .05), n = 2, a = c(1, 30))
sim_n_persons(dexp(1:30, .05), n = 2, a = c(1, 30), ap = c(1,4))
sim_n_persons(dexp(1:30, .05), n = 2, a = 1:5, ap = c(1,1,2,2,3))
```

`sim_n_persons_x_times` *Complete simulation*

Description

Complete simulation

Usage

```
sim_n_persons_x_times(prob, n, a, ap = rep(1/length(a), length(a)),
  times = 100, progress = "text")
```

Arguments

`prob`	Probability to draw a construct from a certain category. Length of vector determines number of categories.
`n`	Number of persons, i.e. grids to sample.
`a`	Number of constructs to be sampled.
`ap`	Probabilities for each number of attributes to be sampled.
`times`	Number of times to repeat each simulation.
`progress`	Type of progress bar shown during simulation.

See Also

Other Simulations: `sim_n_persons_x_times_many_n`, `sim_n_persons`, `sim_one_person`

Examples

```
## Not run:
sim_n_persons_x_times(dexp(1:30, .05), n = 2, a = c(1,30), ap = 1:2, times = 100)
sim_n_persons_x_times(dexp(1:30, .05), n = 2, a = c(1,30), times = 200, progress = "tk")

## End(Not run)
```

`sim_n_persons_x_times_many_n` *Simulate for different n*

Description

Creates simulation results for different n. Runs `sim_n_persons_x_times` for different n.

Usage

```
sim_n_persons_x_times_many_n(prob, n = seq(10, 80, by = 10), a = 7,
  ap = rep(1/length(a), length(a)), times = 100, progress = "text")
```

Arguments

`prob` Probability to draw a construct from a certain category. Length of vector determines number of categories.

`n` Number of persons, i.e. grids to sample.

`a` Number of constructs to be sampled.

`ap` Probabilities for each number of attributes to be sampled.

`times` Number of times to repeat each simulation.

`progress` Type of progress bar shown during simulation.

Value

A result dataframe.

See Also

Other Simulations: `sim_n_persons_x_times`, `sim_n_persons`, `sim_one_person`

Examples

```
## Not run:
r <- sim_n_persons_x_times_many_n(dexp(1:30, .05), a = 7, times = 100)
r <- sim_n_persons_x_times_many_n(dexp(1:30, .05), a = 5:7, ap = 1:3, times = 100)

## End(Not run)
```

`sim_one_person` *Simulate a single grid*

Description

Simulate a single grid

Usage

```
sim_one_person(prob, a = 10)
```

Arguments

`prob` Probability to draw a construct from a certain category.

`a` Number of constructs to be sampled.

See Also

Other Simulations: `sim_n_persons_x_times_many_n`, `sim_n_persons_x_times`, `sim_n_persons`

Examples

```
# draw from exponential distribution
p <- dexp(1:20, rate = .1)
sim_one_person(p, a = 10)
```

Index