

Package ‘stepwedgepower’

October 18, 2026

Type Package

Title Simulation-Based Power Evaluation for Stepped-Wedge Cluster Randomized Trials

Version 0.4.1

Description Provides simulation-based power, type I error, and operating-characteristic evaluation for stepped-wedge cluster randomized trials. General binary-outcome designs support arbitrary crossover schedules, unequal clusters per sequence, secular trends, flexible cluster-period sizes, reproducible parallel execution, sensitivity grids, and mixed-model diagnostics. A component-based engine supports the four treatment states Control, A, B, and A+B under arbitrary asynchronous schedules, including structurally unobserved cluster-periods, withdrawal, reintroduction, wash-in delays, user-defined carryover decay, component main effects, an A-by-B interaction, estimability audits, multiplicity adjustment, conditional and failure-aware power, and explicit classification of hard model errors, nonconvergence, non-finite contrasts, singular fits, and successful fits. A batch-aware interface represents classic and batched stepped-wedge designs with delayed cluster initiation, structurally unobserved cluster-periods, and calendar-time, time-on-trial, or separate batch-specific time effects, following Kasza, Bowden, Hooper and Forbes (2022) <doi:10.1002/sim.9438>. The cumulative Control-to-A-to-A+B interface from earlier versions remains available. The design and mixed-effects analysis follow Hussey and Hughes (2007) <doi:10.1016/j.cct.2006.05.007>. Intraclass correlations for binary outcomes are converted to random-intercept standard deviations on the latent scale of the logistic-normal model, as reviewed by Eldridge, Ukoumunne and Carlin (2009) <doi:10.1111/j.1751-5823.2009.00092.x>. Monte Carlo uncertainty is reported using exact intervals from Clopper and Pearson (1934) <doi:10.1093/biomet/26.4.404>.

URL <https://github.com/AmandaLinLi/stepwedgepower>

BugReports <https://github.com/AmandaLinLi/stepwedgepower/issues>

License MIT + file LICENSE

Encoding UTF-8

Imports lme4, lifecycle, graphics, grDevices, stats, parallel, utils
Suggests testthat (>= 3.0.0), knitr, rmarkdown, swCRTdesign, swdpwr
VignetteBuilder knitr
Config/testthat/edition 3
Config/roxygen2/version 8.1.1
NeedsCompilation no
Author Lin (Amanda) Li [aut, cre],
 Florin Vaida [aut] (degree: PhD)
Maintainer Lin (Amanda) Li <amandali14124277@gmail.com>
Repository CRAN
Date/Publication 2026-10-17 22:40:10 UTC

Contents

analyze_aggregated_outcomes	3
analyze_lpa_outcomes	4
analyze_swcr	5
as_component_design	5
audit_batched_design	6
audit_component_design	7
audit_multistage_design	8
cluster_sd_to_icc	9
compare_batched_designs	9
compare_component_designs	11
compare_multistage_designs	12
compare_with_swdpwr	14
component_resource_summary	15
estimate_group_rates	16
estimate_power	17
estimate_specialty_rates	18
estimate_type1_error	19
fit_batched_model	20
fit_component_model	21
fit_grouped_rate_model	23
fit_multistage_model	24
fit_specialty_rate_model	25
fit_stepwedge_model	26
icc_to_cluster_sd	28
power_batched_swcr	28
power_component_swcr	30
power_grid	31
power_multistage_swcr	32
power_swcr	34
prepare_cluster_data	35
prepare_physician_data	36

read_example_physician_data	37
run_stepwedge_analysis	37
simulate_batched_swcr	38
simulate_component_swcr	38
simulate_multistage_swcr	39
simulate_stepwedge_trial	40
simulate_swcr	41
summarize_by_group	42
summarize_by_specialty	43
sw_assumptions	43
sw_batched_assumptions	45
sw_batched_design	45
sw_component_assumptions	47
sw_component_design	49
sw_component_power	50
sw_design	50
sw_incomplete_component_design	51
sw_multistage_assumptions	52
sw_multistage_design	54
sw_multistage_power	56
sw_power	56
type1_component_swcr	57
type1_multistage_swcr	58
validate_engine	60
Index	61

analyze_aggregated_outcomes

Analyze aggregated grouped outcomes

Description

Generic form of the applied outcome analysis: fits GLM and GLMM rate models for one or more success/trial outcome definitions, across one or more links.

Usage

```
analyze_aggregated_outcomes(
  data,
  cluster = "cluster_id",
  group = "group",
  outcomes,
  links = c("logit", "identity"),
  nAGQ = 10
)
```

Arguments

data	A data frame.
cluster	Cluster identifier column.
group	Name of the grouping column.
outcomes	Named list; each element gives successes and trials.
links	Character vector of links to fit.
nAGQ	Quadrature points.

Value

A nested list of fitted models and group-rate tables.

analyze_lpa_outcomes *Reproduce the core Lp(a) outcome analyses*

Description

Fits the main outcome models from the original script for both overall Lp(a) testing and Lp(a) testing among patients with elevated LDL.

Usage

```
analyze_lpa_outcomes(
  data,
  provider_var = "prov_id",
  specialty_var = "specialty",
  outcomes = list(
    overall = list(successes = "n_lpa_pat", trials = "n_total_pat"),
    high_ldl = list(successes = "n_ldl_lpa_pat", trials = "n_ldl_pat")
  ),
  links = c("logit", "identity"),
  nAGQ = 10
)
```

Arguments

data	A physician-level analysis data frame.
provider_var	Provider identifier column.
specialty_var	Specialty column.
outcomes	Named list defining success and trial columns for each outcome.
links	Character vector of links to fit.
nAGQ	Number of quadrature points for <code>glmer()</code> .

Value

A nested list containing fitted models and specialty-rate tables.

analyze_swcrct	<i>Analyze a simulated stepped-wedge trial</i>
----------------	--

Description

Convenience wrapper around [fit_stepwedge_model()] for data produced by [simulate_swcrct()].

Usage

```
analyze_swcrct(
  trial,
  ...
)
```

Arguments

trial	A data frame from [simulate_swcrct()].
...	Passed on / unused.

Value

The list returned by [fit_stepwedge_model()].

as_component_design	<i>Convert a Cumulative Design to a Component Design</i>
---------------------	--

Description

Maps the version 0.3.0 Control/A/A+B representation to the general four-state component representation.

Usage

```
as_component_design(design)
```

Arguments

design	An sw_multistage_design or sw_component_design object.
--------	--

Value

An sw_component_design object.

Examples

```
old <- sw_multistage_design(c(3, 3), a_start = c(2, 3),
                           b_start = c(4, 5), n_periods = 5)
as_component_design(old)
```

audit_batched_design *Audit a Classic or Batched Stepped-Wedge Design*

Description

Checks batch initiation, observed-state support, fixed-effect rank, and formal estimability under the selected time model.

Usage

```
audit_batched_design(
  design,
  assumptions = NULL,
  time_model = c("calendar", "time_on_trial", "separate"),
  contrasts = NULL,
  include_interaction = FALSE,
  adjust_sequence = FALSE
)
## S3 method for class 'sw_batched_audit'
print(x, ...)
```

Arguments

design	An sw_batched_design object.
assumptions	Optional component or batched assumptions object.
time_model	Planned analysis time parameterization.
contrasts	Requested standard contrasts.
include_interaction	Whether to include the A-by-B interaction. Defaults to FALSE; set to TRUE for adequately supported four-state designs.
adjust_sequence	Whether to include sequence fixed effects.
x	An sw_batched_audit object.
...	Unused.

Value

An object of class sw_batched_audit.

audit_component_design

Audit a Component-Based Stepped-Wedge Design

Description

Checks observed treatment-state support, overlap by calendar period, resource counts, withdrawal and restart patterns, fixed-effect rank, and estimability of requested contrasts. Structurally unobserved cells are excluded from rank and overlap calculations but retained for latent exposure history.

Usage

```
audit_component_design(
  design,
  assumptions = NULL,
  contrasts = NULL,
  include_interaction = TRUE,
  adjust_sequence = FALSE,
  period_effect = c("categorical", "linear", "none")
)

## S3 method for class 'sw_component_audit'
print(x, ...)
```

Arguments

design	An sw_component_design object.
assumptions	Optional sw_component_assumptions object.
contrasts	Requested standard contrasts.
include_interaction	Whether to audit an A-by-B interaction model.
adjust_sequence	Whether to include sequence fixed effects.
period_effect	Categorical, linear, or no period effect.
x	An sw_component_audit object.
...	Unused.

Value

An object of class sw_component_audit.

Examples

```
state <- rbind(S1 = c(0, 1, 3, 2), S2 = c(0, 2, 3, 3),
              S3 = c(0, 1, 1, 3), S4 = c(0, 2, 2, 3))
d <- sw_component_design(rep(3, 4), state = state)
audit_component_design(d)
```

audit_multistage_design

Audit identifiability of an asynchronous cumulative stepped-wedge design

Description

Checks whether a Control → A → A+B schedule provides the concurrent comparisons needed to distinguish intervention effects from calendar time. It also flags two common threats to interpretation: adding B in one common calendar period, and adding B after exactly the same duration of A exposure in every sequence. The latter can make an incremental B effect difficult to distinguish from maturation of A when A itself may change with exposure time.

Usage

```
audit_multistage_design(
  design,
  delay_a = 0L,
  delay_b = 0L,
  adjust_sequence = FALSE
)

## S3 method for class 'sw_multistage_audit'
print(x, ...)
```

Arguments

design	An <code>sw_multistage_design()</code> object.
delay_a, delay_b	Non-negative integer delays used to form the default analysis indicators. These should match the values in <code>sw_multistage_assumptions()</code> .
adjust_sequence	Logical; include sequence fixed effects in the fixed-effect rank check.
x	An <code>sw_multistage_audit</code> object.
...	Unused.

Value

An object of class "sw_multistage_audit" containing overlap periods, rank diagnostics, schedule flags, state counts, and explanatory messages.

Examples

```
design <- sw_multistage_design(
  clusters_per_sequence = rep(4, 4),
  a_start = c(2, 3, 4, 5),
  b_start = c(5, 7, 6, 8),
```

```

    n_periods = 8
  )
  audit_multistage_design(design, delay_a = 1, delay_b = 1)

```

cluster_sd_to_icc	<i>Convert a random-intercept standard deviation to an intraclass correlation</i>
-------------------	---

Description

Inverse of [icc_to_cluster_sd](#).

Usage

```
cluster_sd_to_icc(cluster_sd)
```

Arguments

cluster_sd Numeric ≥ 0 . Random-intercept standard deviation.

Value

The latent-scale intraclass correlation.

Examples

```
cluster_sd_to_icc(0.416)
```

compare_batched_designs	<i>Compare Batched Stepped-Wedge Designs and Time Models</i>
-------------------------	--

Description

Combines operating characteristics, fit diagnostics, time parameterizations, and resource totals for design or time-model comparisons.

Usage

```

compare_batched_designs(
  designs, assumptions, time_models = "calendar", labels = NULL,
  nsim = 1000, alpha = 0.05, contrasts = NULL,
  include_interaction = FALSE,
  multiplicity = c("holm", "bonferroni", "none"),
  multiplicity_family = NULL, fit_link = c("logit", "identity"),
  nAGQ = 1, analysis_args = list(), n_cores = 1, seed = NULL,
  check_design = TRUE, warn_on_design = TRUE
)
compare_swd_bswd(
  swd, bswd, assumptions,
  swd_time_model = "calendar", bswd_time_model = "calendar", ...
)
compare_batched_time_models(
  design, assumptions,
  time_models = c("calendar", "time_on_trial", "separate"), ...
)
## S3 method for class 'sw_batched_comparison'
print(x, ...)

```

Arguments

<code>designs</code>	List of at least two <code>sw_batched_design</code> objects.
<code>assumptions</code>	One batched assumptions object or one per design.
<code>time_models</code>	One analysis time model or one per design.
<code>labels</code>	Optional design labels.
<code>nsim</code> , <code>alpha</code> , <code>contrasts</code> , <code>include_interaction</code> , <code>multiplicity</code> , <code>multiplicity_family</code> , <code>fit_link</code> , <code>nAGQ</code> , <code>analysis_args</code> , <code>n_cores</code> , <code>seed</code> , <code>check_design</code> , <code>warn_on_design</code>	Arguments passed to <code>power_batched_swcr</code> .
<code>swd</code> , <code>bswd</code>	Classic and batched design objects.
<code>swd_time_model</code> , <code>bswd_time_model</code>	Analysis time models for the two designs.
<code>design</code>	One batched design for a time-model comparison.
<code>x</code>	An <code>sw_batched_comparison</code> object.
<code>...</code>	Additional arguments or unused.

Value

An object of class `sw_batched_comparison`.

compare_component_designs

Compare Component-Based Stepped-Wedge Designs

Description

Runs the component power engine for multiple schedules and reports operating characteristics and resource totals side by side. Calendar and observed cluster-period totals are reported separately so complete and batched stepped-wedge designs can be compared under equal observed resources.

Usage

```
compare_component_designs(
  designs,
  assumptions,
  labels = NULL,
  nsim = 1000,
  alpha = 0.05,
  contrasts = NULL,
  include_interaction = TRUE,
  multiplicity = c("holm", "bonferroni", "none"),
  multiplicity_family = NULL,
  fit_link = c("logit", "identity"),
  nAGQ = 1,
  analysis_args = list(),
  n_cores = 1,
  seed = NULL,
  check_design = TRUE,
  warn_on_design = TRUE
)

## S3 method for class 'sw_component_comparison'
print(x, ...)
```

Arguments

designs	A list of at least two sw_component_design objects.
assumptions	One assumptions object used for all designs, or one per design.
labels	Optional design labels.
nsim, alpha, contrasts, include_interaction, multiplicity, multiplicity_family, fit_link, nAGQ, analysis_args, n_cores, seed, check_design, warn_on_design	Arguments passed to power_component_swcrct.
x	An sw_component_comparison object.
...	Unused.

Value

An object of class `sw_component_comparison`. The `fit_diagnostics` element combines the five fit-category counts and rates across designs.

Examples

```
state <- rbind(S1 = c(0, 1, 3, 2), S2 = c(0, 2, 3, 3),
               S3 = c(0, 1, 1, 3), S4 = c(0, 2, 2, 3))
d <- sw_component_design(rep(3, 4), state = state)
a <- sw_component_assumptions(baseline_prob = 0.15,
                             treatment_or_a = 1.5, treatment_or_b = 1.4,
                             interaction_or = 1.1, icc = 0.05)
compare_component_designs(list(first = d, second = d), a,
                          nsim = 10, seed = 1,
                          warn_on_design = FALSE)
```

`compare_multistage_designs`

Compare two-state and cumulative three-state stepped-wedge designs

Description

Runs `power_multistage_swcrf()` for two candidate schedules, commonly a Control → A design and an asynchronous Control → A → A+B design. The function reports the operating characteristics side by side and checks whether the designs use the same number of clusters, periods, and cluster-period observations.

Usage

```
compare_multistage_designs(
  design_two,
  design_three,
  assumptions_two,
  assumptions_three = assumptions_two,
  labels = c("Control → A", "Control → A → A+B"),
  nsim = 1000,
  alpha = 0.05,
  multiplicity = c("holm", "bonferroni", "none"),
  fit_link = c("logit", "identity"),
  nAGQ = 1,
  analysis_args = list(),
  n_cores = 1,
  seed = NULL,
  check_design = TRUE,
  warn_on_design = TRUE
)
```

```
## S3 method for class 'sw_multistage_comparison'
print(x, ...)
```

Arguments

design_two, design_three	Two <code>sw_multistage_design()</code> objects. The first would normally have <code>b_start = Inf</code> for every sequence.
assumptions_two	A <code>sw_multistage_assumptions()</code> object for the first design.
assumptions_three	Assumptions for the second design. Defaults to <code>assumptions_two</code> .
labels	Two display labels.
nsim	Number of simulated trials.
alpha	Two-sided significance level.
multiplicity	Adjustment for the A and incremental-B component tests: "holm", "bonferroni", or "none".
fit_link	Link used in <code>fit_multistage_model()</code> .
nAGQ	Adaptive Gauss-Hermite quadrature points.
analysis_args	Optional named list of additional arguments passed to <code>fit_multistage_model()</code> , such as <code>adjust_sequence</code> or <code>period_effect</code> .
n_cores	Number of cores. Reproducibility under a fixed seed does not depend on this value.
seed	Optional random seed.
check_design	Logical; audit the schedule before simulation and stop for a rank-deficient or non-estimable default model.
warn_on_design	Logical; emit non-fatal design-audit cautions.
x	An <code>sw_multistage_comparison</code> object.
...	Unused.

Details

When the same seed is used, both designs start from the same reproducible simulation streams. This can reduce simulation noise in a like-for-like comparison when their dimensions match.

Value

An object of class "sw_multistage_comparison" containing both power objects, a stacked comparison table, a combined fit-diagnostics table, and a resource check.

Examples

```
three <- sw_multistage_design(
  clusters_per_sequence = rep(4, 4),
  a_start = c(2, 3, 4, 5), b_start = c(5, 7, 6, 8),
  n_periods = 8
)
two <- sw_multistage_design(
  clusters_per_sequence = rep(4, 4),
  a_start = c(2, 3, 4, 5), b_start = rep(Inf, 4),
  n_periods = 8
)
a <- sw_multistage_assumptions(
  baseline_prob = 0.15, treatment_or_a = 1.4,
  incremental_or_b = 1.3, delay_a = 1, delay_b = 1,
  icc = 0.05, n_per_cluster_period = 30
)
compare_multistage_designs(two, three, a, nsim = 100, seed = 1)
```

compare_with_swdpwr	<i>Informational comparison with a closed-form power package</i>
---------------------	--

Description

Runs [power_swcr](#) alongside the closed-form calculation in the **swdpwr** package. Offered for information, not as a validation target: closed-form stepped-wedge power is asymptotic in the number of clusters and can be markedly optimistic at the small cluster counts these trials typically use.

Usage

```
compare_with_swdpwr(
  clusters_per_sequence,
  n_per_cluster_period,
  baseline_prob,
  treatment_or,
  icc,
  alpha = 0.05,
  nsim = 500,
  seed = NULL,
  n_cores = 1,
  adjust_sequence = FALSE
)
```

Arguments

`clusters_per_sequence`
Integer vector; must be constant across sequences for this comparison.

n_per_cluster_period	Cluster-period size (a single value).
baseline_prob	Common baseline probability under control.
treatment_or	Treatment odds ratio.
icc	Latent-scale intraclass correlation.
alpha	Two-sided significance level.
nsim	Number of simulations per grid cell.
seed	Optional seed.
n_cores	Number of cores. 1 runs serially. Results are reproducible under a given seed regardless of n_cores.
adjust_sequence	Whether the fitted model includes a fixed sequence term.

Value

A one-row data frame with the simulated power, its Monte Carlo interval, the closed-form power, and their difference.

References

Chen J, Zhou X, Li F, Spiegelman D (2022). "swdpwr: A SAS macro and an R package for power calculations in stepped wedge cluster randomized trials." *Computer Methods and Programs in Biomedicine*, 213, 106522. doi:[10.1016/j.cmpb.2021.106522](https://doi.org/10.1016/j.cmpb.2021.106522)

Zhou X, Liao X, Kunz LM, Normand SLT, Wang M, Spiegelman D (2020). "A maximum likelihood approach to power calculations for stepped wedge designs of binary outcomes." *Biostatistics*, 21(1), 102-121. doi:[10.1093/biostatistics/kxy031](https://doi.org/10.1093/biostatistics/kxy031)

component_resource_summary

Summarize Resources for a Component-Based Design

Description

Reports calendar and observed sequence-periods and cluster-periods. Structurally unobserved cells are omitted from observed-resource totals.

Usage

```
component_resource_summary(design, assumptions = NULL)
```

Arguments

design	An sw_component_design object.
assumptions	Optional sw_component_assumptions object. When supplied, total individual observations are calculated from the cluster-period sample-size specification.

Value

A one-row data frame containing calendar, observed, and missing resource counts and, when available, total individual observations.

Examples

```
design <- sw_incomplete_component_design(
  c(2, 2, 3, 3),
  rbind(
    c("0", "1", "1+2", "1+2", "-"),
    c("0", "0", "1", "1+2", "-"),
    c("-", "0", "1", "1+2", "1+2"),
    c("-", "0", "0", "1", "1+2")
  )
)
component_resource_summary(design)
```

estimate_group_rates	<i>Estimate group-specific probabilities from a fitted rate model</i>
----------------------	---

Description

Extracts group-level probabilities from a model produced by [fit_grouped_rate_model](#).

Usage

```
estimate_group_rates(
  model,
  group_levels = NULL,
  group = "group",
  link = c("logit", "identity"),
  approximate_marginal = TRUE,
  logit_scale_factor = 0.346
)
```

Arguments

model	A fitted rate model.
group_levels	Optional vector of group levels.
group	Name of the grouping column.
link	Link function.
approximate_marginal	Logical; apply the marginal approximation.
logit_scale_factor	Approximation constant.

Value

A data frame of group-level linear predictors and probabilities.

estimate_power	<i>Estimate power by repeated stepped-wedge simulation</i>
----------------	--

Description

Repeats the stepped-wedge simulation-and-analysis workflow and estimates power.

Usage

```
estimate_power(
  n_simulations = 100,
  alpha = 0.05,
  treatment_or = NULL,
  n_clusters_per_sequence = NULL,
  sequence_names = NULL,
  cluster_sd = NULL,
  icc = NULL,
  baseline_probs = NULL,
  n_per_cluster_period = NULL,
  n_steps = NULL,
  fit_link = c("logit", "identity"),
  seed = NULL,
  nAGQ = 1,
  effect_size_or = NULL,
  n_providers_per_specialty = NULL,
  specialty_names = NULL,
  base_probs = NULL,
  tau_provider = NULL,
  pts_per_step = NULL
)
```

Arguments

n_simulations	Number of simulations.
alpha	Significance threshold.
treatment_or	Odds ratio under the data-generating model.
n_clusters_per_sequence	Integer vector of clusters per sequence.
sequence_names	Labels for the sequences.
cluster_sd	Standard deviation of the cluster random intercept.
icc	Optional intraclass correlation (alternative to cluster_sd).
baseline_probs	Baseline probabilities by sequence.

n_per_cluster_period	Units per cluster per period.
n_steps	Number of periods.
fit_link	Link used when fitting the analysis model.
seed	Optional seed.
nAGQ	Number of quadrature points.
effect_size_or	Deprecated legacy alias for treatment_or.
n_providers_per_specialty	Deprecated legacy alias for n_clusters_per_sequence.
specialty_names	Deprecated legacy alias for sequence_names.
base_probs	Deprecated legacy alias for baseline_probs.
tau_provider	Deprecated legacy alias for cluster_sd.
pts_per_step	Deprecated legacy alias for n_per_cluster_period.

Value

A list with the estimated power and the vector of p-values.

estimate_specialty_rates

Estimate specialty-specific probabilities from a fitted model

Description

Extracts specialty-level probabilities from a fitted specialty model.

Usage

```
estimate_specialty_rates(
  model,
  specialty_levels = NULL,
  specialty_var = "specialty",
  link = c("logit", "identity"),
  approximate_marginal = TRUE,
  logit_scale_factor = 0.346
)
```

Arguments

model	A fitted model returned by fit_specialty_rate_model .
specialty_levels	Optional vector of specialty levels.
specialty_var	Name of the specialty column used in the model.

link Link function for the fitted model.
 approximate_marginal Logical; whether to apply the random-intercept logit approximation.
 logit_scale_factor Approximation constant used in the shrinkage factor.

Value

A data frame with specialty-level linear predictors and probabilities.

estimate_type1_error *Estimate type I error by repeated stepped-wedge simulation*

Description

A convenience wrapper around [estimate_power](#) that sets the treatment odds ratio to 1.

Usage

```
estimate_type1_error(
  n_simulations = 100,
  alpha = 0.05,
  n_clusters_per_sequence = NULL,
  sequence_names = NULL,
  cluster_sd = NULL,
  icc = NULL,
  baseline_probs = NULL,
  n_per_cluster_period = NULL,
  n_steps = NULL,
  fit_link = c("logit", "identity"),
  seed = NULL,
  nAGQ = 1,
  n_providers_per_specialty = NULL,
  specialty_names = NULL,
  base_probs = NULL,
  tau_provider = NULL,
  pts_per_step = NULL
)
```

Arguments

n_simulations Number of simulations.
 alpha Significance threshold.
 n_clusters_per_sequence Integer vector of clusters per sequence.
 sequence_names Labels for the sequences.

cluster_sd	Standard deviation of the cluster random intercept.
icc	Optional intraclass correlation (alternative to cluster_sd).
baseline_probs	Baseline probabilities by sequence.
n_per_cluster_period	Units per cluster per period.
n_steps	Number of periods.
fit_link	Link used when fitting the analysis model.
seed	Optional seed.
nAGQ	Number of quadrature points.
n_providers_per_specialty	Deprecated legacy alias for n_clusters_per_sequence.
specialty_names	Deprecated legacy alias for sequence_names.
base_probs	Deprecated legacy alias for baseline_probs.
tau_provider	Deprecated legacy alias for cluster_sd.
pts_per_step	Deprecated legacy alias for n_per_cluster_period.

Value

A list like [estimate_power](#) with the type I error estimate added.

fit_batched_model	<i>Fit a Classic or Batched Stepped-Wedge Analysis Model</i>
-------------------	--

Description

Fits the component GLMM with shared categorical calendar effects, shared categorical time-on-trial effects, or a categorical batch-by-local-time factor giving each batch a separate time profile.

Usage

```
fit_batched_model(
  data,
  time_model = c("calendar", "time_on_trial", "separate"),
  formula = NULL,
  outcome = c("events", "n"),
  contrasts = NULL,
  include_interaction = FALSE,
  multiplicity = c("holm", "bonferroni", "none"),
  multiplicity_family = NULL,
  adjust_sequence = FALSE,
  link = c("logit", "identity"),
  nAGQ = 1,
  alpha = 0.05
)
```

Arguments

data	Aggregated cluster-period data from <code>simulate_batched_swcr</code> .
time_model	Calendar, time-on-trial, or separate time parameterization.
formula	Optional custom mixed-model formula.
outcome	Names of event and total-count columns.
contrasts	Standard component contrasts.
include_interaction	Whether to fit the A-by-B interaction. Defaults to FALSE, which is appropriate for cumulative Control/A/A+B schedules without a B-only state.
multiplicity	Multiplicity adjustment.
multiplicity_family	Contrasts in the adjustment family.
adjust_sequence	Whether to include sequence fixed effects.
link	Binomial link.
nAGQ	Adaptive quadrature points.
alpha	Two-sided significance level.

Value

The result from `fit_component_model` with batched-time metadata.

<code>fit_component_model</code>	<i>Fit a Component-Based Stepped-Wedge Model</i>
----------------------------------	--

Description

Fits the planned mixed model and returns A versus Control, B versus Control, B versus A, combined-treatment, incremental, and interaction Wald contrasts with covariance-correct standard errors. If the data contain an observed column, rows marked false or zero are removed before outcome validation and fitting.

Usage

```
fit_component_model(
  data,
  formula = NULL,
  outcome = c("events", "n"),
  weight_a = "a_effect_weight",
  weight_b = "b_effect_weight",
  weight_ab = "ab_effect_weight",
  term_a = weight_a,
  term_b = weight_b,
```

```

term_ab = weight_ab,
include_interaction = TRUE,
contrasts = NULL,
multiplicity = c("holm", "bonferroni", "none"),
multiplicity_family = NULL,
period = "period",
cluster = "cluster_id",
sequence = "sequence",
adjust_sequence = FALSE,
period_effect = c("categorical", "linear", "none"),
link = c("logit", "identity"),
nAGQ = 1,
alpha = 0.05
)

```

Arguments

data	Aggregated cluster-period data.
formula	Optional custom mixed-model formula.
outcome	Names of event and total-count columns.
weight_a, weight_b, weight_ab	Predictor names for effect weights.
term_a, term_b, term_ab	Fitted coefficient names used in contrasts.
include_interaction	Whether to include an interaction term.
contrasts	Standard contrasts to report.
multiplicity	None, Holm, or Bonferroni adjustment.
multiplicity_family	Contrasts included in the adjustment family.
period, cluster, sequence	Column names.
adjust_sequence	Whether to include sequence fixed effects.
period_effect	Categorical, linear, or no period effect.
link	Binomial link.
nAGQ	Adaptive quadrature points.
alpha	Two-sided significance level.

Value

A list containing the fitted model, contrast table, multiplicity results, and fitting diagnostics. The `fit_status` field distinguishes `hard_glm_error`, `nonconverged_fit`, `converged_nonfinite_contrast`, and `successful_fit`. Singularity is returned separately in `singular`; a singular fit is not automatically classified as unsuccessful. Error, optimizer, convergence-message, and non-finite-contrast details are retained.

Examples

```

state <- rbind(S1 = c(0, 1, 3, 2), S2 = c(0, 2, 3, 3),
              S3 = c(0, 1, 1, 3), S4 = c(0, 2, 2, 3))
d <- sw_component_design(rep(4, 4), state = state)
a <- sw_component_assumptions(baseline_prob = 0.15,
                             treatment_or_a = 1.6, treatment_or_b = 1.5,
                             interaction_or = 1.1, icc = 0.05, n_per_cluster_period = 40)
fit_component_model(simulate_component_swcrct(d, a, seed = 1))$contrasts

```

fit_grouped_rate_model

Fit a grouped rate model

Description

Fits a binomial GLM or cluster-random-intercept GLMM for an aggregated success/trial outcome grouped by an arbitrary grouping variable.

Usage

```

fit_grouped_rate_model(
  data,
  successes,
  trials,
  group = "group",
  cluster = NULL,
  link = c("logit", "identity"),
  random_intercept = !is.null(cluster),
  nAGQ = 10
)

```

Arguments

data	A data frame.
successes	Name of the success-count column.
trials	Name of the trial-count column.
group	Name of the grouping column.
cluster	Cluster identifier column.
link	Link function.
random_intercept	Logical; include a cluster random intercept.
nAGQ	Quadrature points.

Value

A fitted glm or merMod object.

fit_multistage_model *Fit a cumulative A then A+B stepped-wedge analysis model*

Description

Fits a binomial cluster-random-intercept model with separate indicators for the delayed effect of A and the delayed incremental effect of B. The default model is

Usage

```
fit_multistage_model(
  data,
  formula = NULL,
  outcome = c("events", "n"),
  intervention_a = "a_effective",
  intervention_b = "b_effective",
  term_a = intervention_a,
  term_b = intervention_b,
  period = "period",
  cluster = "cluster_id",
  sequence = "sequence",
  adjust_sequence = FALSE,
  period_effect = c("categorical", "linear", "none"),
  link = c("logit", "identity"),
  nAGQ = 1,
  alpha = 0.05,
  multiplicity = c("holm", "bonferroni", "none")
)
```

Arguments

data	Aggregated cluster-period data, typically from simulate_multistage_swcrf() .
formula	Optional custom model formula. If supplied, the variable-name and structural arguments are ignored, but term_a and term_b must name the fitted coefficients representing A and incremental B.
outcome	Length-two character vector c(events, n).
intervention_a	Name of the delayed A indicator.
intervention_b	Name of the delayed B indicator.
term_a, term_b	Coefficient names used to construct contrasts. By default they equal intervention_a and intervention_b.
period	Name of the calendar-period column.
cluster	Name of the cluster identifier.
sequence	Name of the sequence column.
adjust_sequence	Logical; include a fixed sequence effect.

period_effect	One of "categorical", "linear", or "none".
link	Binomial link, "logit" or "identity".
nAGQ	Adaptive Gauss-Hermite quadrature points.
alpha	Two-sided significance level used for confidence intervals and rejection indicators.
multiplicity	Adjustment for the two component tests A and B. One of "none", "holm", or "bonferroni". The total A+B contrast is reported separately and is not included in this adjustment family.

Details

`cbind(events, n - events) ~ a_effective + b_effective + factor(period) + (1 | cluster_id).`

The A coefficient compares A with control after the A wash-in period. The B coefficient compares A+B with A after the B wash-in period. The total A+B effect relative to control is the linear contrast $\beta_A + \beta_B$ and its standard error includes the covariance between the two estimates.

Value

A list containing the fitted model, a contrast table, raw and adjusted p-values, detailed fit status, convergence and singularity diagnostics, captured warnings, and a joint-success indicator for the A and B component tests. Fit status distinguishes hard GLMM errors, nonconvergence, converged fits with non-finite contrasts, and successful fits; singularity is reported separately.

Examples

```
design <- sw_multistage_design(
  clusters_per_sequence = rep(5, 4),
  a_start = c(2, 3, 4, 5), b_start = c(5, 7, 6, 8),
  n_periods = 8
)
assumptions <- sw_multistage_assumptions(
  baseline_prob = 0.15, treatment_or_a = 1.6,
  incremental_or_b = 1.4, delay_a = 1, delay_b = 1,
  icc = 0.05, n_per_cluster_period = 40
)
fit <- fit_multistage_model(
  simulate_multistage_swcrct(design, assumptions, seed = 1)
)
fit$contrasts
```

fit_specialty_rate_model

Fit a specialty-level testing-rate model

Description

Fits either a binomial GLM or a provider-random-intercept binomial GLMM for aggregated success/trial data.

Usage

```
fit_specialty_rate_model(
  data,
  successes,
  trials,
  specialty_var = "specialty",
  provider_var = NULL,
  link = c("logit", "identity"),
  random_intercept = !is.null(provider_var),
  nAGQ = 10
)
```

Arguments

<code>data</code>	A data frame containing counts and grouping variables.
<code>successes</code>	Name of the success-count column.
<code>trials</code>	Name of the trial-count column.
<code>specialty_var</code>	Name of the specialty column.
<code>provider_var</code>	Optional provider identifier column.
<code>link</code>	Link function. Supported values are "logit" and "identity".
<code>random_intercept</code>	Logical; whether to include a provider random intercept.
<code>nAGQ</code>	Number of adaptive Gauss-Hermite quadrature points for <code>lme4::glmer()</code> .

Value

A fitted `glm` or `merMod` object.

<code>fit_stepwedge_model</code>	<i>Fit a stepped-wedge analysis model</i>
----------------------------------	---

Description

Fits a binomial random-intercept model to aggregated cluster-period data, configurable through variable names or a user-supplied formula. Independent of any application-specific column names.

Usage

```
fit_stepwedge_model(
  data,
  formula = NULL,
  outcome = c("events", "n"),
  treatment = "intervention",
  period = "period",
  cluster = "cluster_id",
  sequence = "sequence",
  adjust_sequence = FALSE,
  period_effect = c("categorical", "linear", "none"),
  link = c("logit", "identity"),
  nAGQ = 1
)
```

Arguments

data	Aggregated cluster-period data.
formula	Optional model formula; if supplied, the naming/structural arguments are ignored.
outcome	Outcome type. Currently "binary".
treatment	Optional 0/1 matrix (sequences x periods); overrides crossover_period.
period	Name of the period column.
cluster	Name of the cluster identifier column.
sequence	Name of the sequence column.
adjust_sequence	Logical; include a fixed sequence adjustment.
period_effect	One of "categorical", "linear", "none".
link	Binomial link, "logit" or "identity".
nAGQ	Number of adaptive Gauss-Hermite quadrature points.

Value

A list with the fitted model, treatment coefficient row, estimate, standard error, p-value, and convergence/singularity flags.

Examples

```
d <- sw_design(c(6,6,6,6), crossover_period = c(2,3,4,5), n_periods = 5)
a <- sw_assumptions(baseline_prob = 0.1, treatment_or = 2, icc = 0.03)
fit_stepwedge_model(simulate_swcrct(d, a, seed = 1))$p_value
```

icc_to_cluster_sd	<i>Convert an intraclass correlation to a random-intercept standard deviation</i>
-------------------	---

Description

For a logistic random-intercept model the latent-scale residual variance is $\pi^2/3$. This maps a latent-scale ICC to the cluster-level random-intercept standard deviation.

Usage

```
icc_to_cluster_sd(icc)
```

Arguments

`icc` Numeric in $[0, 1)$. Intraclass correlation on the latent scale.

Value

The random-intercept standard deviation.

Examples

```
icc_to_cluster_sd(0.05)
```

power_batched_swcr	<i>Power and Type I Error for Batched Stepped-Wedge Designs</i>
--------------------	---

Description

Estimates operating characteristics under a selected analysis time model. The generating time model remains separately specified in the assumptions object, allowing matched or misspecified time-model studies.

Usage

```
power_batched_swcr(
  design, assumptions,
  analysis_time_model = c("calendar", "time_on_trial", "separate"),
  nsim = 1000, alpha = 0.05, contrasts = NULL,
  include_interaction = FALSE,
  multiplicity = c("holm", "bonferroni", "none"),
  multiplicity_family = NULL, fit_link = c("logit", "identity"),
  nAGQ = 1, analysis_args = list(), n_cores = 1, seed = NULL,
  check_design = TRUE, warn_on_design = TRUE
)
```

```

type1_batched_swcr(
  design, assumptions,
  analysis_time_model = c("calendar", "time_on_trial", "separate"),
  nsim = 1000, alpha = 0.05, contrasts = NULL,
  include_interaction = FALSE,
  multiplicity = c("holm", "bonferroni", "none"),
  multiplicity_family = NULL, fit_link = c("logit", "identity"),
  nAGQ = 1, analysis_args = list(), n_cores = 1, seed = NULL,
  check_design = TRUE, warn_on_design = TRUE
)
## S3 method for class 'sw_batched_power'
print(x, ...)

```

Arguments

<code>design</code>	An <code>sw_batched_design</code> object.
<code>assumptions</code>	An <code>sw_batched_assumptions</code> object.
<code>analysis_time_model</code>	Time parameterization used in the fitted model.
<code>nsim</code>	Number of simulated trials.
<code>alpha</code>	Two-sided significance level.
<code>contrasts</code>	Standard component contrasts.
<code>include_interaction</code>	Whether to fit the A-by-B interaction. Defaults to FALSE; set to TRUE for adequately supported four-state designs.
<code>multiplicity</code>	Multiplicity adjustment.
<code>multiplicity_family</code>	Contrasts in the adjustment family.
<code>fit_link</code>	Binomial analysis link.
<code>nAGQ</code>	Adaptive quadrature points.
<code>analysis_args</code>	Additional arguments passed to the component fitting engine.
<code>n_cores</code>	Number of processor cores.
<code>seed</code>	Optional reproducibility seed.
<code>check_design</code>	Whether to run a batched-design audit.
<code>warn_on_design</code>	Whether audit cautions produce a warning.
<code>x</code>	An <code>sw_batched_power</code> object.
<code>...</code>	Unused.

Value

An object of class `sw_batched_power`, inheriting from `sw_component_power`.

power_component_swcrtr *Estimate power for a component-based stepped-wedge design*

Description

Repeatedly simulates a Control/A/B/A+B trial and fits the planned component mixed model. The output distinguishes conditional power among evaluable fits from failure-aware power, which counts unusable analyses as unsuccessful trials. It also reports estimation bias, standard-error calibration, confidence-interval coverage, convergence, singularity, and Monte Carlo uncertainty for every requested contrast. Fit diagnostics separately report hard GLMM errors, nonconverged fits, converged fits with non-finite requested contrasts, singular fits, and successful fits. Singularity is a non-exclusive flag and does not automatically make a fit unsuccessful.

Usage

```
power_component_swcrtr(
  design,
  assumptions,
  nsim = 1000,
  alpha = 0.05,
  contrasts = NULL,
  include_interaction = TRUE,
  multiplicity = c("holm", "bonferroni", "none"),
  multiplicity_family = NULL,
  fit_link = c("logit", "identity"),
  nAGQ = 1,
  analysis_args = list(),
  n_cores = 1,
  seed = NULL,
  check_design = TRUE,
  warn_on_design = TRUE
)
```

Arguments

design	An <code>sw_component_design()</code> object.
assumptions	An <code>sw_component_assumptions()</code> object.
nsim	Number of simulated trials.
alpha	Two-sided significance level.
contrasts	Standard contrasts to evaluate.
include_interaction	Logical; fit the interaction term.
multiplicity	Multiplicity adjustment for multiplicity_family.
multiplicity_family	Contrasts forming the multiplicity family.

fit_link	Binomial link used by <code>fit_component_model()</code> .
nAGQ	Quadrature points for the analysis model.
analysis_args	Optional named list of additional arguments passed to <code>fit_component_model()</code> .
n_cores	Number of processor cores. Reproducibility under a fixed seed does not depend on this value.
seed	Optional random seed.
check_design	Logical; run <code>audit_component_design()</code> first.
warn_on_design	Logical; warn when the audit returns cautions.

Value

An object of class "sw_component_power". In addition to the power and estimation tables, `fit_diagnostics` gives aggregate counts and rates for the five fit categories, and `replicate_diagnostics` gives one row per simulation with error and convergence messages.

Examples

```
state <- rbind(
  S1 = c(0, 1, 1, 3, 3, 2),
  S2 = c(0, 0, 2, 2, 3, 3),
  S3 = c(0, 0, 1, 1, 3, 3),
  S4 = c(0, 0, 2, 3, 3, 3)
)
d <- sw_component_design(rep(4, 4), state = state)
a <- sw_component_assumptions(
  baseline_prob = 0.15, treatment_or_a = 1.5,
  treatment_or_b = 1.4, interaction_or = 1.1,
  carryover_periods_a = 1, carryover_weights_a = 0.5,
  icc = 0.05, n_per_cluster_period = 30
)
p <- power_component_swcrtr(d, a, nsim = 20, seed = 1,
  warn_on_design = FALSE)
p
```

power_grid

Power across a grid of design and assumption values

Description

Runs `power_swcrtr` across the Cartesian product of one or more varying parameters, holding everything else fixed. Supports parallel execution, with results reproducible under a given seed regardless of the number of cores.

Usage

```
power_grid(
  design,
  assumptions,
  vary,
  nsim = 500,
  alpha = 0.05,
  n_cores = 1,
  seed = NULL,
  ...
)
```

Arguments

design	An sw_design object.
assumptions	An sw_assumptions object.
vary	A named list of vectors to vary. Names may be any sw_assumptions argument and/or clusters_per_sequence, which must then be a list of integer vectors.
nsim	Number of simulations per grid cell.
alpha	Two-sided significance level.
n_cores	Number of cores. 1 runs serially. Results are reproducible under a given seed regardless of n_cores.
seed	Optional seed.
...	Passed to power_swcr .

Value

An object of class "sw_power_grid": a data frame with one row per grid cell, carrying the full sw_power objects in an attribute.

power_multistage_swcr

Estimate power for an asynchronous Control to A to A+B design

Description

Repeatedly simulates a cumulative-intervention stepped-wedge trial and fits the prespecified mixed model. Separate operating characteristics are reported for A versus Control, incremental B (A+B versus A), total A+B versus Control, multiplicity-adjusted component tests, and joint success of both component tests.

Usage

```
power_multistage_swcr(
  design,
  assumptions,
  nsim = 1000,
  alpha = 0.05,
  multiplicity = c("holm", "bonferroni", "none"),
  fit_link = c("logit", "identity"),
  nAGQ = 1,
  analysis_args = list(),
  n_cores = 1,
  seed = NULL,
  check_design = TRUE,
  warn_on_design = TRUE
)
```

Arguments

design	An <code>sw_multistage_design()</code> object.
assumptions	An <code>sw_multistage_assumptions()</code> object.
nsim	Number of simulated trials.
alpha	Two-sided significance level.
multiplicity	Adjustment for the A and incremental-B component tests: "holm", "bonferroni", or "none".
fit_link	Link used in <code>fit_multistage_model()</code> .
nAGQ	Adaptive Gauss-Hermite quadrature points.
analysis_args	Optional named list of additional arguments passed to <code>fit_multistage_model()</code> , such as <code>adjust_sequence</code> or <code>period_effect</code> .
n_cores	Number of cores. Reproducibility under a fixed seed does not depend on this value.
seed	Optional random seed.
check_design	Logical; audit the schedule before simulation and stop for a rank-deficient or non-estimable default model.
warn_on_design	Logical; emit non-fatal design-audit cautions.

Details

Two power definitions are reported. **Conditional power** uses only replicates in which the planned analysis converged and produced the relevant test. **Failure-aware power** divides the same number of rejections by all simulated trials, treating an unusable analysis as an unsuccessful trial. Their difference reveals how much nominal performance depends on silently discarding failed fits.

Value

An object of class "sw_multistage_power". The `fit_diagnostics` element reports hard GLMM errors, nonconverged fits, converged fits with non-finite contrasts, singular fits, and successful fits. The `replicate_diagnostics` element retains one row per simulation.

Examples

```

design <- sw_multistage_design(
  clusters_per_sequence = rep(5, 5),
  a_start = c(2, 3, 4, 5, 6),
  b_start = c(6, 8, 7, 9, 10),
  n_periods = 10
)
assumptions <- sw_multistage_assumptions(
  baseline_prob = 0.15,
  treatment_or_a = 1.4,
  incremental_or_b = 1.3,
  delay_a = 1, delay_b = 1,
  icc = 0.05,
  n_per_cluster_period = 30
)
result <- power_multistage_swcr(
  design, assumptions, nsim = 100, seed = 1
)
result

```

power_swcr

Estimate power for a stepped-wedge trial by simulation

Description

Repeatedly simulates from an [sw_design()] and [sw_assumptions()], fits the analysis model, and estimates rejection probability at level alpha. Returns an [sw_power] object with the Monte Carlo standard error, a confidence interval for power, per-simulation estimates and standard errors, convergence and singular-fit diagnostics, and empirical bias and coverage.

Usage

```

power_swcr(
  design,
  assumptions,
  nsim = 1000,
  alpha = 0.05,
  fit_link = c("logit", "identity"),
  nAGQ = 1,
  analysis_args = list(),
  n_cores = 1,
  seed = NULL
)

```

Arguments

design	An [sw_design()] object.
assumptions	An [sw_assumptions()] object.
nsim	Number of simulations.
alpha	Two-sided significance level.
fit_link	Link used when fitting the analysis model.
nAGQ	Number of adaptive Gauss-Hermite quadrature points.
analysis_args	Optional named list of extra arguments to [fit_stepwedge_model()].
n_cores	Number of cores. 1 runs serially. Results are reproducible under a given seed regardless of n_cores, because each replicate is assigned its own L'Ecuyer-CMRG random-number stream.
seed	Optional random seed.

Value

An object of class "sw_power".

Examples

```
d <- sw_design(c(8,8,8,8), crossover_period = c(2,3,4,5), n_periods = 5)
a <- sw_assumptions(baseline_prob = 0.05, treatment_or = 2, icc = 0.03)
summary(power_swcrf(d, a, nsim = 50, seed = 1))
```

prepare_cluster_data *Prepare grouped cluster-level analysis data*

Description

Filters a cluster-level dataset to the groups of interest, keeps clusters within a size window, and sorts the result.

Usage

```
prepare_cluster_data(
  data,
  groups = NULL,
  min_size = 0,
  max_size = Inf,
  group_var = "group",
  size_var = "n",
  order_var = NULL
)
```

Arguments

data	A data frame.
groups	Optional character vector of groups to keep.
min_size	Minimum cluster size (inclusive).
max_size	Maximum cluster size (exclusive).
group_var	Name of the grouping column.
size_var	Name of the cluster-size column.
order_var	Optional ordering column.

Value

A filtered and sorted data frame.

```
prepare_physician_data
```

Prepare physician-level stepped-wedge analysis data

Description

Filters the input data to the specialties of interest, applies panel-size thresholds, removes extreme outliers, and sorts the output.

Usage

```
prepare_physician_data(
  data,
  specialties = c("CARDIOLOGY", "FAMILY MEDICINE", "INTERNAL MEDICINE", "NEUROLOGY"),
  min_patients = 100,
  max_patients = 10000,
  specialty_var = "specialty",
  patient_var = "n_total_pat",
  provider_name_var = "PROV_NAME"
)
```

Arguments

data	A data frame.
specialties	Character vector of specialties to keep.
min_patients	Minimum total number of patients required.
max_patients	Maximum total number of patients allowed.
specialty_var	Name of the specialty column.
patient_var	Name of the total-patient count column.
provider_name_var	Name of the provider name column used for ordering.

Value

A filtered and sorted data frame.

```
read_example_physician_data
```

Read the bundled example physician data

Description

Reads a small synthetic physician-level example dataset bundled with the package.

Usage

```
read_example_physician_data()
```

Value

A data frame.

```
run_stepwedge_analysis
```

Fit the stepped-wedge analysis model to a simulated dataset

Description

Fits the mixed-effects stepped-wedge analysis model used in the original script.

Usage

```
run_stepwedge_analysis(
  sim_data,
  fit_link = c("logit", "identity"),
  nAGQ = 1
)
```

Arguments

sim_data	A data frame from <code>simulate_stepwedge_trial()</code> or a compatible aggregated cluster-period dataset.
fit_link	Link used when fitting the analysis model.
nAGQ	Number of quadrature points.

Value

A list with the fitted model, coefficient table, and treatment p-value.

 simulate_batched_swcr

Simulate a Classic or Batched Stepped-Wedge Trial

Description

Simulates aggregated binary cluster-period outcomes and appends batch, calendar-period, and time-on-trial variables.

Usage

```
simulate_batched_swcr(design, assumptions, seed = NULL, include_unobserved = FALSE)
```

Arguments

design	An sw_batched_design object.
assumptions	An sw_batched_assumptions object.
seed	Optional random seed.
include_unobserved	Whether to retain latent unobserved rows with missing outcomes.

Value

A data frame of cluster-period outcomes and treatment-history variables.

 simulate_component_swcr

Simulate a Component-Based Stepped-Wedge Trial

Description

Generates aggregated binary cluster-period outcomes under arbitrary Control/A/B/A+B schedules with wash-in, withdrawal, carryover, restart, secular trend, component effects, interaction, and structurally unobserved periods.

Usage

```
simulate_component_swcr(
  design,
  assumptions,
  seed = NULL,
  include_unobserved = FALSE
)
```

Arguments

design	An <code>sw_component_design</code> object.
assumptions	An <code>sw_component_assumptions</code> object.
seed	Optional random seed.
include_unobserved	Logical. When false, return only observed cluster-periods. When true, retain latent unobserved rows with missing n and events.

Value

A data frame containing assignments, exposure histories, effect weights, outcomes, and true probabilities.

Examples

```
state <- rbind(S1 = c(0, 1, 3, 2), S2 = c(0, 2, 3, 3))
d <- sw_component_design(c(3, 3), state = state)
a <- sw_component_assumptions(baseline_prob = 0.15,
  treatment_or_a = 1.4, treatment_or_b = 1.3,
  interaction_or = 1.1, icc = 0.05)
head(simulate_component_swcr(d, a, seed = 1))
```

`simulate_multistage_swcr`

Simulate an asynchronous Control to A to A+B stepped-wedge trial

Description

Generates aggregated cluster-period binary outcomes for a cumulative intervention design. Sequences can start A and add B in different periods, and each intervention can have a prespecified wash-in delay.

Usage

```
simulate_multistage_swcr(design, assumptions, seed = NULL)
```

Arguments

design	An <code>sw_multistage_design()</code> object.
assumptions	An <code>sw_multistage_assumptions()</code> object.
seed	Optional random seed.

Value

A data frame with one row per cluster-period. In addition to the standard count columns, it contains the active treatment state, exposure times since A and B began, and the delayed-effect indicators used by the default analysis.

Examples

```

design <- sw_multistage_design(
  clusters_per_sequence = rep(4, 4),
  a_start = c(2, 3, 4, 5),
  b_start = c(5, 7, 6, 8),
  n_periods = 8
)
assumptions <- sw_multistage_assumptions(
  baseline_prob = 0.15, treatment_or_a = 1.4,
  incremental_or_b = 1.3, delay_a = 1, delay_b = 1,
  icc = 0.05, n_per_cluster_period = 30
)
head(simulate_multistage_swcrct(design, assumptions, seed = 1))

```

simulate_stepwedge_trial

Simulate one stepped-wedge trial dataset

Description

Generates aggregated provider-by-step binomial data for a sequential stepped-wedge design.

Usage

```

simulate_stepwedge_trial(
  treatment_or = NULL,
  n_clusters_per_sequence = NULL,
  sequence_names = NULL,
  baseline_probs = NULL,
  cluster_sd = NULL,
  icc = NULL,
  n_per_cluster_period = NULL,
  n_steps = NULL,
  seed = NULL,
  effect_size_or = NULL,
  n_providers_per_specialty = NULL,
  specialty_names = NULL,
  base_probs = NULL,
  tau_provider = NULL,
  pts_per_step = NULL
)

```

Arguments

treatment_or Odds ratio under the data-generating model.
n_clusters_per_sequence
 Integer vector of clusters per sequence.

sequence_names	Labels for the sequences.
baseline_probs	Baseline probabilities by sequence.
cluster_sd	Standard deviation of the cluster random intercept.
icc	Optional intraclass correlation (alternative to cluster_sd).
n_per_cluster_period	Units per cluster per period.
n_steps	Number of periods.
seed	Optional seed.
effect_size_or	Deprecated legacy alias for treatment_or.
n_providers_per_specialty	Deprecated legacy alias for n_clusters_per_sequence.
specialty_names	Deprecated legacy alias for sequence_names.
base_probs	Deprecated legacy alias for baseline_probs.
tau_provider	Deprecated legacy alias for cluster_sd.
pts_per_step	Deprecated legacy alias for n_per_cluster_period.

Value

A data frame with one row per provider-step combination.

simulate_swcr	<i>Simulate one stepped-wedge cluster-randomized trial</i>
---------------	--

Description

Generates an aggregated cluster-by-period dataset from an [sw_design()] and [sw_assumptions()] under a logistic random-intercept model with treatment, period, and baseline effects. Application-agnostic: clusters may be hospitals, clinics, schools, practices, physicians, or regions.

Usage

```
simulate_swcr(
  design,
  assumptions,
  seed = NULL
)
```

Arguments

design	An [sw_design()] object.
assumptions	An [sw_assumptions()] object.
seed	Optional random seed.

Value

A data frame with one row per cluster-period (columns `cluster_id`, `sequence`, `sequence_idx`, `period`, `intervention`, `n`, `events`, `true_prob`).

Examples

```
d <- sw_design(c(8,8,8,8), crossover_period = c(2,3,4,5), n_periods = 5)
a <- sw_assumptions(baseline_prob = 0.05, treatment_or = 1.5, icc = 0.05)
head(simulate_swcrct(d, a, seed = 1))
```

summarize_by_group	<i>Summarize numeric variables by group</i>
--------------------	---

Description

Computes the sample size and summary statistics of one or more numeric variables within each level of a grouping variable.

Usage

```
summarize_by_group(
  data,
  group_var = "group",
  vars,
  na.rm = TRUE
)
```

Arguments

<code>data</code>	A data frame.
<code>group_var</code>	Name of the grouping column.
<code>vars</code>	Character vector of numeric variable names.
<code>na.rm</code>	Logical; remove missing values.

Value

A data frame with one row per group-variable combination.

summarize_by_specialty	<i>Summarize physician counts by specialty</i>
------------------------	--

Description

Computes common summary statistics for one or more numeric variables within each specialty.

Usage

```
summarize_by_specialty(  
  data,  
  specialty_var = "specialty",  
  vars = c("n_total_pat", "n_ldl_pat"),  
  na.rm = TRUE  
)
```

Arguments

- data A data frame.
- specialty_var Name of the specialty column.
- vars Character vector of numeric variable names to summarize.
- na.rm Logical; whether to remove missing values.

Value

A data frame with one row per specialty-variable combination.

sw_assumptions	<i>Specify the data-generating assumptions for a stepped-wedge trial</i>
----------------	--

Description

Collects baseline risk, treatment effect, cluster random-effect magnitude (via `icc` or `cluster_sd`), secular period trend, and cluster-period sample size for simulation. Baseline and sample size may be scalar, per-sequence, or per sequence-period.

Usage

```
sw_assumptions(
  outcome = "binary",
  baseline_prob = NULL,
  baseline_logit = NULL,
  treatment_or = NULL,
  treatment_effect = NULL,
  icc = NULL,
  cluster_sd = NULL,
  period_effects = NULL,
  n_per_cluster_period = 20
)
```

Arguments

outcome	Outcome type. Currently "binary".
baseline_prob	Baseline probability: scalar, length-n_sequences vector, or n_sequences x n_periods matrix.
baseline_logit	Baseline linear predictor on the logit scale (same shapes as baseline_prob).
treatment_or	Treatment odds ratio.
treatment_effect	Treatment effect on the log-odds scale.
icc	Latent-scale intraclass correlation.
cluster_sd	Random-intercept standard deviation.
period_effects	Length-n_periods secular trend on the log-odds scale.
n_per_cluster_period	Cluster-period sample size: scalar, n_sequences x n_periods matrix, or a function of the number of rows.

Value

An object of class "sw_assumptions".

Examples

```
sw_assumptions(baseline_prob = 0.05, treatment_or = 1.5, icc = 0.05,
  period_effects = c(0,0.05,0.1,0.15,0.2), n_per_cluster_period = 20)
```

sw_batched_assumptions

Specify Assumptions for a Batched Stepped-Wedge Trial

Description

Extends component assumptions with one of the three time parameterizations used for batched stepped-wedge designs.

Usage

```
sw_batched_assumptions(
  ...,
  time_model = c("calendar", "time_on_trial", "separate"),
  time_effects = NULL
)

## S3 method for class 'sw_batched_assumptions'
print(x, ...)
```

Arguments

...	Arguments passed to sw_component_assumptions. Do not supply period_effects; use time_effects.
time_model	Data-generating time parameterization: calendar time, shared time on trial, or separate batch-specific time.
time_effects	Time-effect profile on the log-odds scale. For separate time, supply a numeric matrix or list with one profile per batch.
x	An sw_batched_assumptions object.

Value

An object of class sw_batched_assumptions, inheriting from sw_component_assumptions.

sw_batched_design

Construct a Classic or Batched Stepped-Wedge Design

Description

Creates a batch-aware component design. A batched stepped-wedge design groups sequences into batches that begin trial participation at staggered calendar periods. Dashes are structurally unobserved cluster-periods and are not Control observations.

Usage

```
sw_batched_design(
  clusters_per_sequence,
  state,
  batch = NULL,
  latent_state = NULL,
  sequence_names = NULL,
  missing = c("-", "--", "-", "-", "", "NA", ".")
)

as_batched_design(design, batch = NULL)

## S3 method for class 'sw_batched_design'
print(x, ...)
```

Arguments

clusters_per_sequence	Positive integer vector giving clusters per sequence.
state	Sequence-by-calendar-period treatment schedule. Labels may include 0, 1, 2, 1+2, Control, A, B, and A+B.
batch	Optional batch label per sequence. When NULL, batches are inferred from first observed calendar periods.
latent_state	Optional complete latent treatment-state matrix for schedules with internal unobserved cells.
sequence_names	Optional sequence labels.
missing	Character markers treated as structurally unobserved.
design	An sw_component_design object.
x	An sw_batched_design object.
...	Unused.

Details

All sequences in a batch must share the same first observed calendar period. The returned object records calendar period, time on trial, batch starts, delays from the first batch, gaps from the preceding batch, cluster totals by batch, and the treatment schedule and observation mask.

Value

An object of class sw_batched_design, inheriting from sw_component_design.

Examples

```
bswd <- sw_batched_design(
  c(2, 2, 3, 3),
  rbind(
    c("0", "1", "1+2", "1+2", "-"),
```

```

      c("0", "0", "1", "1+2", "-"),
      c("-", "0", "1", "1+2", "1+2"),
      c("-", "0", "0", "1", "1+2")
    )
  )
  bswd

```

sw_component_assumptions

Specify Assumptions for a Component-Based Trial

Description

Defines component main effects, interaction, wash-in, withdrawal carryover, restart rules, correlation, secular trend, and sample size for the four-state engine.

Usage

```

sw_component_assumptions(
  outcome = "binary",
  baseline_prob = NULL,
  baseline_logit = NULL,
  treatment_or_a = NULL,
  treatment_effect_a = NULL,
  treatment_or_b = NULL,
  treatment_effect_b = NULL,
  interaction_or = NULL,
  interaction_effect = NULL,
  delay_a = 0L,
  delay_b = 0L,
  carryover_periods_a = 0L,
  carryover_periods_b = 0L,
  carryover_weights_a = NULL,
  carryover_weights_b = NULL,
  restart_rule_a = c("reset", "resume"),
  restart_rule_b = c("reset", "resume"),
  interaction_mode = c("effective_overlap", "assigned_overlap", "none"),
  icc = NULL,
  cluster_sd = NULL,
  period_effects = NULL,
  n_per_cluster_period = 20
)

## S3 method for class 'sw_component_assumptions'
print(x, ...)

```

Arguments

outcome Currently only "binary".
 baseline_prob, baseline_logit Supply exactly one baseline probability or baseline logit specification.
 treatment_or_a, treatment_effect_a A main effect as an odds ratio or log-odds coefficient.
 treatment_or_b, treatment_effect_b B main effect.
 interaction_or, interaction_effect A-by-B interaction as an odds ratio or log-odds coefficient.
 delay_a, delay_b Non-negative wash-in periods.
 carryover_periods_a, carryover_periods_b Periods of residual effect after withdrawal.
 carryover_weights_a, carryover_weights_b Non-increasing weights in [0, 1], one per carryover period or one repeated value.
 restart_rule_a, restart_rule_b Whether a restarted component resets or resumes its wash-in clock.
 interaction_mode Whether interaction follows effective overlap, current assigned overlap, or is suppressed.
 icc, cluster_sd Supply one latent-scale ICC or cluster random-intercept standard deviation.
 period_effects Optional secular log-odds effects.
 n_per_cluster_period Cluster-period sample size specification.
 x An sw_component_assumptions object.
 ... Unused.

Value

An object of class sw_component_assumptions.

Examples

```

a <- sw_component_assumptions(
  baseline_prob = 0.15,
  treatment_or_a = 1.35,
  treatment_or_b = 1.25,
  interaction_or = 1.10,
  carryover_periods_a = 2,
  carryover_weights_a = c(0.5, 0.25),
  icc = 0.05
)
a

```

sw_component_design *Construct a Component-Based Stepped-Wedge Design*

Description

Creates an arbitrary asynchronous schedule containing Control, A, B, and A+B. The schedule may include withdrawal, reintroduction, and structurally unobserved cluster-periods, and need not be monotone. Use `sw_incomplete_component_design()` for schedules written directly with dashes or missing values.

Usage

```
sw_component_design(
  clusters_per_sequence,
  state = NULL,
  component_a = NULL,
  component_b = NULL,
  sequence_names = NULL,
  observed = NULL
)

## S3 method for class 'sw_component_design'
print(x, ...)
```

Arguments

<code>clusters_per_sequence</code>	Positive integer vector giving the number of clusters in each sequence.
<code>state</code>	Optional sequence-by-period matrix containing codes 0 (Control), 1 (A), 2 (B), and 3 (A+B), or equivalent character labels such as "1+2".
<code>component_a</code> , <code>component_b</code>	Optional binary sequence-by-period assignment matrices. Supply both when <code>state</code> is omitted.
<code>sequence_names</code>	Optional unique sequence labels.
<code>observed</code>	Optional logical or 0/1 sequence-by-period matrix. A false value marks a structurally unobserved cell. Its latent treatment state is retained for exposure history but it contributes no outcome or fitted-model row.
<code>x</code>	An <code>sw_component_design</code> object.
<code>...</code>	Unused.

Value

An object of class `sw_component_design`.

Examples

```

state <- rbind(
  S1 = c("Control", "A", "A", "A+B", "A+B", "B"),
  S2 = c("Control", "Control", "B", "B", "A+B", "A+B"),
  S3 = c("Control", "Control", "A", "A", "A+B", "A+B")
)
design <- sw_component_design(c(4, 4, 4), state = state)
design

```

sw_component_power	<i>Methods for component-based power objects</i>
--------------------	--

Description

Methods for component-based power objects

Usage

```

## S3 method for class 'sw_component_power'
print(x, ...)

## S3 method for class 'sw_component_power'
summary(object, ...)

## S3 method for class 'sw_component_power'
plot(x, type = c("conditional", "failure-aware"), ...)

```

Arguments

x, object	An sw_component_power object.
...	Additional graphical arguments for plot; otherwise unused.
type	Plot "conditional" or "failure-aware" rejection rates.

sw_design	<i>Construct a stepped-wedge design</i>
-----------	---

Description

Builds a stepped-wedge design object with a 0/1 treatment matrix (sequences x periods), specified either by per-sequence crossover periods or by an explicit treatment matrix. Supports multiple baseline periods, unequal clusters per sequence, and custom or simultaneous crossovers.

Usage

```
sw_design(
  clusters_per_sequence,
  crossover_period = NULL,
  n_periods = NULL,
  treatment = NULL,
  sequence_names = NULL
)
```

Arguments

`clusters_per_sequence` Integer vector; number of clusters in each sequence.

`crossover_period` Integer vector; period at which each sequence turns on (Inf = never). Ignored if treatment is given.

`n_periods` Number of periods; inferred if NULL.

`treatment` Optional 0/1 matrix (sequences x periods); overrides `crossover_period`.

`sequence_names` Optional sequence labels.

Value

An object of class "sw_design".

Examples

```
sw_design(clusters_per_sequence = c(10,10,10,10),
  crossover_period = c(2,3,4,5), n_periods = 5)
```

sw_incomplete_component_design

Construct a Component Design with Structurally Missing Periods

Description

Creates a component-based stepped-wedge design from a manuscript-style schedule in which dashes, blanks, or missing values mean that no outcome is observed. Leading missing cells inherit the first observed state and trailing missing cells inherit the last observed state. The inferred latent state is used only to preserve wash-in, withdrawal, carryover, and restart history; unobserved cells are excluded from simulation outcomes, model fitting, power, and resource totals.

Usage

```
sw_incomplete_component_design(
  clusters_per_sequence,
  state,
  latent_state = NULL,
  sequence_names = NULL,
  missing = c("-", "--", "-", "-", "", "NA", ".")
)
```

Arguments

clusters_per_sequence	Positive integer vector giving the number of clusters in each sequence.
state	Sequence-by-period schedule containing treatment labels and structurally missing markers. Labels may include 0, 1, 2, and 1+2.
latent_state	Optional complete treatment-state matrix defining the latent state in every calendar period, including unobserved cells. It is required for internal missing gaps whose latent history cannot be inferred.
sequence_names	Optional unique sequence labels.
missing	Character values interpreted as structurally missing cells rather than Control. Actual NA values are always missing.

Value

An object of class `sw_component_design` with an observation mask.

Examples

```
bswd <- sw_incomplete_component_design(
  c(2, 2, 3, 3),
  rbind(
    c("0", "1", "1+2", "1+2", "-"),
    c("0", "0", "1", "1+2", "-"),
    c("-", "0", "1", "1+2", "1+2"),
    c("-", "0", "0", "1", "1+2")
  )
)
bswd
```

sw_multistage_assumptions

Specify assumptions for a cumulative A then A+B stepped-wedge trial

Description

Collects the data-generating assumptions for an asynchronous cumulative intervention design created by `sw_multistage_design()`. The binary-outcome generating model is

Usage

```
sw_multistage_assumptions(
  outcome = "binary",
  baseline_prob = NULL,
  baseline_logit = NULL,
  treatment_or_a = NULL,
  treatment_effect_a = NULL,
  incremental_or_b = NULL,
  incremental_effect_b = NULL,
  delay_a = 0L,
  delay_b = 0L,
  icc = NULL,
  cluster_sd = NULL,
  period_effects = NULL,
  n_per_cluster_period = 20
)

## S3 method for class 'sw_multistage_assumptions'
print(x, ...)
```

Arguments

outcome	Outcome type. Currently "binary".
baseline_prob	Baseline probability: a scalar, a vector with one value per sequence, or a sequence-by-period matrix. Supply this or baseline_logit, not both.
baseline_logit	Baseline linear predictor on the logit scale, with the same permitted shapes as baseline_prob.
treatment_or_a	Odds ratio for A versus control after the A delay. Supply this or treatment_effect_a.
treatment_effect_a	Effect of A on the log-odds scale.
incremental_or_b	Incremental odds ratio for adding B to A after the B delay. Supply this or incremental_effect_b. Defaults to one.
incremental_effect_b	Incremental B effect on the log-odds scale.
delay_a, delay_b	Non-negative integer wash-in periods for A and B.
icc	Latent-scale intraclass correlation. Supply this or cluster_sd.
cluster_sd	Cluster random-intercept standard deviation.
period_effects	Numeric vector with one secular log-odds effect per period. NULL means no secular trend.
n_per_cluster_period	Cluster-period sample size: a scalar, a sequence-by-period matrix, or a function returning one count per row.
x	An sw_multistage_assumptions object.
...	Unused.

Details

$$\text{logit}(p_{ijt}) = \beta_{0,jt} + \gamma_t + b_i + \beta_A A_{ijt}^* + \beta_B B_{ijt}^*,$$

where A* becomes one after A has completed its prespecified delay and B* becomes one after B has completed its delay. Because B is never delivered alone, $\exp(\beta_B)$ is the incremental effect of adding B among clusters already receiving A. The total A+B effect relative to control is $\exp(\beta_A + \beta_B)$.

A delay of zero means that the effect begins in the first active period. A delay of one treats the first active period as wash-in and begins the effect in the second exposure period.

Value

An object of class "sw_multistage_assumptions".

Examples

```
assumptions <- sw_multistage_assumptions(
  baseline_prob = 0.15,
  treatment_or_a = 1.40,
  incremental_or_b = 1.30,
  delay_a = 1,
  delay_b = 1,
  icc = 0.05,
  n_per_cluster_period = 30
)
assumptions
```

sw_multistage_design	<i>Construct an asynchronous cumulative-intervention stepped-wedge design</i>
----------------------	---

Description

Builds a design in which clusters move through the ordered treatment states Control → A → A+B. Sequences may start intervention A and add intervention B in different calendar periods. B is therefore interpreted as an add-on intervention delivered only after A has begun; the design contains no B-only condition.

Usage

```
sw_multistage_design(
  clusters_per_sequence,
  a_start = NULL,
  b_start = NULL,
  n_periods = NULL,
  state = NULL,
```

```

    sequence_names = NULL
  )

  ## S3 method for class 'sw_multistage_design'
  print(x, ...)

```

Arguments

clusters_per_sequence	Positive integer vector giving the number of clusters assigned to each sequence.
a_start	Integer vector giving the first period under intervention A for each sequence. Use Inf for a sequence that never starts A. Ignored when state is supplied.
b_start	Integer vector giving the first period under A+B for each sequence. Each finite value must be strictly later than the corresponding a_start. Use Inf for a sequence that never adds B. Ignored when state is supplied.
n_periods	Number of periods. If NULL, inferred from the largest finite start period or the number of columns in state.
state	Optional integer matrix with values 0, 1, and 2, one row per sequence and one column per period. This overrides a_start and b_start.
sequence_names	Optional unique labels for the sequences.
x	An sw_multistage_design object.
...	Unused.

Details

The design can be specified either by sequence-specific start periods or by an explicit state matrix. In a state matrix, rows are sequences, columns are periods, and entries are 0 (Control), 1 (A), or 2 (A+B). Every row must be non-decreasing and must contain at least one A-only period before A+B.

Value

An object of class "sw_multistage_design".

Examples

```

design <- sw_multistage_design(
  clusters_per_sequence = rep(6, 5),
  a_start = c(2, 3, 4, 5, 6),
  b_start = c(6, 7, 8, 9, 10),
  n_periods = 10
)
design

```

sw_multistage_power	<i>Methods for multistage stepped-wedge power objects</i>
---------------------	---

Description

Methods for multistage stepped-wedge power objects

Usage

```
## S3 method for class 'sw_multistage_power'
print(x, ...)

## S3 method for class 'sw_multistage_power'
summary(object, ...)

## S3 method for class 'sw_multistage_power'
plot(x, type = c("conditional", "failure-aware"), ...)
```

Arguments

x, object	An sw_multistage_power object.
...	Additional graphical arguments for plot; otherwise unused.
type	Power definition for the plot: "conditional" or "failure-aware".

sw_power	<i>Methods for sw_power objects</i>
----------	-------------------------------------

Description

print, summary, and plot methods for the object returned by [power_swcrct](#). plot shows the distribution of estimated treatment effects with the true effect and mean estimate marked.

Usage

```
## S3 method for class 'sw_power'
print(x, ...)

## S3 method for class 'sw_power'
summary(object, ...)

## S3 method for class 'sw_power'
plot(x, ...)
```

Arguments

x, object	An sw_power object from power_swcr .
...	Unused.

Value

Invisibly, the sw_power object. Called for their printing / plotting side effects.

type1_component_swcr *Estimate global-null type I error for a component design*

Description

Sets the A, B, and interaction effects to zero while preserving the schedule, carryover structure, ICC, secular trend, sample size, and planned analysis.

Usage

```
type1_component_swcr(
  design,
  assumptions,
  nsim = 1000,
  alpha = 0.05,
  contrasts = NULL,
  include_interaction = TRUE,
  multiplicity = c("holm", "bonferroni", "none"),
  multiplicity_family = NULL,
  fit_link = c("logit", "identity"),
  nAGQ = 1,
  analysis_args = list(),
  n_cores = 1,
  seed = NULL,
  check_design = TRUE,
  warn_on_design = TRUE
)
```

Arguments

design	An sw_component_design() object.
assumptions	An sw_component_assumptions() object.
nsim	Number of simulated trials.
alpha	Two-sided significance level.
contrasts	Standard contrasts to evaluate.
include_interaction	Logical; fit the interaction term.

multiplicity Multiplicity adjustment for multiplicity_family.
multiplicity_family
 Contrasts forming the multiplicity family.
fit_link Binomial link used by `fit_component_model()`.
nAGQ Quadrature points for the analysis model.
analysis_args Optional named list of additional arguments passed to `fit_component_model()`.
n_cores Number of processor cores. Reproducibility under a fixed seed does not depend on this value.
seed Optional random seed.
check_design Logical; run `audit_component_design()` first.
warn_on_design Logical; warn when the audit returns cautions.

Value

An `sw_component_power` object evaluated under the global null.

Examples

```

state <- rbind(
  S1 = c(0, 1, 1, 3), S2 = c(0, 2, 2, 3),
  S3 = c(0, 0, 1, 3), S4 = c(0, 0, 2, 3)
)
d <- sw_component_design(rep(3, 4), state = state)
a <- sw_component_assumptions(
  baseline_prob = 0.15, treatment_or_a = 1.4,
  treatment_or_b = 1.3, interaction_or = 1.1, icc = 0.05
)
type1_component_swcr(d, a, nsim = 20, seed = 1,
  warn_on_design = FALSE)

```

type1_multistage_swcr

Estimate global-null type I error for a multistage stepped-wedge design

Description

Convenience wrapper around `power_multistage_swcr()` that sets both the A and incremental-B effects to zero while preserving every other assumption.

Usage

```

type1_multistage_swcr(
  design,
  assumptions,
  nsim = 1000,
  alpha = 0.05,
  multiplicity = c("holm", "bonferroni", "none"),
  fit_link = c("logit", "identity"),
  nAGQ = 1,
  analysis_args = list(),
  n_cores = 1,
  seed = NULL,
  check_design = TRUE,
  warn_on_design = TRUE
)

```

Arguments

design	An <code>sw_multistage_design()</code> object.
assumptions	An <code>sw_multistage_assumptions()</code> object.
nsim	Number of simulated trials.
alpha	Two-sided significance level.
multiplicity	Adjustment for the A and incremental-B component tests: "holm", "bonferroni", or "none".
fit_link	Link used in <code>fit_multistage_model()</code> .
nAGQ	Adaptive Gauss-Hermite quadrature points.
analysis_args	Optional named list of additional arguments passed to <code>fit_multistage_model()</code> , such as <code>adjust_sequence</code> or <code>period_effect</code> .
n_cores	Number of cores. Reproducibility under a fixed seed does not depend on this value.
seed	Optional random seed.
check_design	Logical; audit the schedule before simulation and stop for a rank-deficient or non-estimable default model.
warn_on_design	Logical; emit non-fatal design-audit cautions.

Value

An `sw_multistage_power` object evaluated under the global null.

Examples

```

design <- sw_multistage_design(
  clusters_per_sequence = rep(4, 4),
  a_start = c(2, 3, 4, 5), b_start = c(5, 7, 6, 8),
  n_periods = 8
)

```

```
a <- sw_multistage_assumptions(
  baseline_prob = 0.15, treatment_or_a = 1.4,
  incremental_or_b = 1.3, icc = 0.05
)
type1_multistage_swcrct(design, a, nsim = 100, seed = 1)
```

 validate_engine

Validate the simulation engine against an independent reference

Description

Cross-checks `power_swcrct` against a reference implementation that shares none of its code, and reports whether the two agree within Monte Carlo error. This is a self-contained check and deliberately does not treat any external power package as ground truth.

Usage

```
validate_engine(
  design,
  assumptions,
  nsim = 500,
  alpha = 0.05,
  seed = NULL,
  tolerance = 0.05
)
```

Arguments

design	An <code>sw_design</code> object.
assumptions	An <code>sw_assumptions</code> object.
nsim	Number of simulations per grid cell.
alpha	Two-sided significance level.
seed	Optional seed.
tolerance	Maximum acceptable absolute difference in power beyond combined Monte Carlo error.

Value

A one-row data frame comparing the engine's power with the reference's.

Index

analyze_aggregated_outcomes, 3
analyze_lpa_outcomes, 4
analyze_swcrct, 5
as_batched_design (sw_batched_design), 45
as_component_design, 5
audit_batched_design, 6
audit_component_design, 7
audit_component_design(), 31, 58
audit_multistage_design, 8

cluster_sd_to_icc, 9
compare_batched_designs, 9
compare_batched_time_models (compare_batched_designs), 9
compare_component_designs, 11
compare_multistage_designs, 12
compare_swd_bswd (compare_batched_designs), 9
compare_with_swdpwr, 14
component_resource_summary, 15

estimate_group_rates, 16
estimate_power, 17, 19, 20
estimate_specialty_rates, 18
estimate_type1_error, 19

fit_batched_model, 20
fit_component_model, 21
fit_component_model(), 31, 58
fit_grouped_rate_model, 16, 23
fit_multistage_model, 24
fit_multistage_model(), 13, 33, 59
fit_specialty_rate_model, 18, 25
fit_stepwedge_model, 26

icc_to_cluster_sd, 9, 28

plot.sw_component_power (sw_component_power), 50

plot.sw_multistage_power (sw_multistage_power), 56
plot.sw_power (sw_power), 56
plot.sw_power_grid (power_grid), 31
power_batched_swcrct, 28
power_component_swcrct, 30
power_grid, 31
power_multistage_swcrct, 32
power_multistage_swcrct(), 12, 58
power_swcrct, 14, 31, 32, 34, 56, 57, 60
prepare_cluster_data, 35
prepare_physician_data, 36
print.sw_assumptions (sw_assumptions), 43
print.sw_batched_assumptions (sw_batched_assumptions), 45
print.sw_batched_audit (audit_batched_design), 6
print.sw_batched_comparison (compare_batched_designs), 9
print.sw_batched_design (sw_batched_design), 45
print.sw_batched_power (power_batched_swcrct), 28
print.sw_component_assumptions (sw_component_assumptions), 47
print.sw_component_audit (audit_component_design), 7
print.sw_component_comparison (compare_component_designs), 11
print.sw_component_design (sw_component_design), 49
print.sw_component_power (sw_component_power), 50
print.sw_design (sw_design), 50
print.sw_multistage_assumptions (sw_multistage_assumptions), 52
print.sw_multistage_audit (audit_multistage_design), 8

print.sw_multistage_comparison
 (compare_multistage_designs),
 12
 print.sw_multistage_design
 (sw_multistage_design), 54
 print.sw_multistage_power
 (sw_multistage_power), 56
 print.sw_power(sw_power), 56
 print.sw_power_grid(power_grid), 31

 read_example_physician_data, 37
 run_stepwedge_analysis, 37

 simulate_batched_swcr, 38
 simulate_component_swcr, 38
 simulate_multistage_swcr, 39
 simulate_multistage_swcr(), 24
 simulate_stepwedge_trial, 40
 simulate_swcr, 41
 summarize_by_group, 42
 summarize_by_specialty, 43
 summary.sw_component_power
 (sw_component_power), 50
 summary.sw_multistage_power
 (sw_multistage_power), 56
 summary.sw_power(sw_power), 56
 summary.sw_power_grid(power_grid), 31
 sw_assumptions, 32, 43, 60
 sw_batched_assumptions, 45
 sw_batched_design, 45
 sw_component_assumptions, 47
 sw_component_assumptions(), 30, 57
 sw_component_design, 49
 sw_component_design(), 30, 57
 sw_component_power, 50
 sw_design, 32, 50, 60
 sw_incomplete_component_design, 51
 sw_multistage_assumptions, 52
 sw_multistage_assumptions(), 8, 13, 33,
 39, 59
 sw_multistage_design, 54
 sw_multistage_design(), 8, 13, 33, 39, 52,
 59
 sw_multistage_power, 56
 sw_power, 56

 type1_batched_swcr
 (power_batched_swcr), 28
 type1_component_swcr, 57

 type1_multistage_swcr, 58
 validate_engine, 60