

Package: PhysioTrial (via r-universe)

October 3, 2026

Type Package

Title Randomization, Allocation Concealment and Blinding for Clinical Trials

Version 0.5.1

Description Provides reproducible trial-design infrastructure for simple, permuted-block, stratified-block, and Pocock-Simon minimization randomization. Sealed allocation sequences, deterministic audit fingerprints, treatment-code management, append-only unblinding logs, CONSORT/SPIRIT operations, adverse-event summaries, explicit analysis populations, missing-data hooks, and endpoint-analysis wrappers support auditable clinical-trial workflows. Deterministic CDISC SDTM-shaped and ADaM-shaped exports provide linked tabulation and analysis datasets without claiming external regulatory conformance.

Depends R (>= 4.2.0)

Imports digest, methods, PhysioExperiment (>= 2.0.0), stats

Suggests PhysioCore, DiagrammeR, PhysioClinical, PhysioClinStats, knitr, rmarkdown, testthat (>= 3.2.0)

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Maintainer Yusuke Matsui <mail.to.matsui@gmail.com>

URL <https://github.com/x-biosignal/PhysioTrial>,
<https://x-biosignal.r-universe.dev/PhysioTrial>,
<https://x-biosignal.github.io/PhysioTrial/>

BugReports <https://github.com/x-biosignal/PhysioTrial/issues>

Config/testthat/edition 3

VignetteBuilder knitr

Collate 'PhysioTrial-package.R' 'class-trial.R' 'randomization.R'
 'blinding.R' 'consort.R' 'spirit.R' 'adverse-events.R'
 'analysis-sets.R' 'endpoints.R' 'power.R' 'cdisc.R' 'zzz.R'

Config/pak/sysreqs zlib1g-dev

Repository <https://x-biosignal.r-universe.dev>

Date/Publication 2026-10-03 13:11:44 UTC

RemoteUrl <https://github.com/x-biosignal/PhysioTrial>

RemoteRef HEAD

RemoteSha 1d209055a7a05ebdf94e80651a149907d47da782

Contents

adverseEvent	3
aeSummary	4
allocationRatio	5
allocationTable	6
analysisData	6
analysisSetAccessors	7
Arm	8
Arm-class	9
arms	9
assignments	10
auditLog	10
blindingCodes	11
blindingIndex	11
blindingManager	12
BlindingManager-class	13
consortCounts	13
consortDiagram	14
defineXML	15
endpointANCOVA	16
endpointMMRM	17
estimateBaselineCorrelation	18
intentionToTreat	19
nextAllocation	20
Participant	20
Participant-class	21
perProtocol	21
powerBinary	23
powerContinuous	23
powerCurve	24
RandomizationSequence-class	25
randomize	25
randomSeed	26
responderAnalysis	27
sampleSizeANCOVA	28

sampleSizeBinary	29
sampleSizeChangeScore	30
sampleSizeContinuous	30
sampleSizeRepeatedMeasures	32
spiritChecklist	33
spiritSchedule	33
strata	34
toADaM	35
toSDTM	36
Trial	37
Trial-class	38
trialFingerprint	38
unblind	39
unblindingLog	39
validateCDISC	40
Index	41

adverseEvent	<i>Capture an Adverse Event</i>
--------------	---------------------------------

Description

Creates one typed adverse-event row. CTCAE severity and regulatory seriousness are stored independently; a serious event is not inferred from a grade threshold. MedDRA fields preserve externally assigned coding metadata and are not checked against a bundled terminology.

Usage

```
adverseEvent(  
  participant_id,  
  term,  
  onset_date,  
  end_date = as.Date(NA),  
  ctcae_grade,  
  serious = FALSE,  
  causality = c("not_assessed", "unrelated", "unlikely", "possible", "probable",  
    "definite"),  
  action = NA_character_,  
  outcome = c("unknown", "not_recovered", "recovering", "recovered",  
    "recovered_with_sequelae", "fatal"),  
  meddra_pt = NA_character_,  
  meddra_code = NA_character_,  
  meddra_version = NA_character_,  
  metadata = list()  
)
```

Arguments

participant_id	Non-empty participant identifier.
term	Verbatim adverse-event term.
onset_date	Event onset date.
end_date	Optional event end date.
ctcae_grade	Integer CTCAE grade from 1 through 5.
serious	Whether the event meets a seriousness criterion.
causality	Investigator causality assessment.
action	Optional action taken.
outcome	Controlled event outcome.
meddra_pt	Optional MedDRA preferred term.
meddra_code	Optional MedDRA code.
meddra_version	Optional MedDRA version.
metadata	Free-form list stored as a list-column.

Value

A one-row adverse_event data frame.

References

National Cancer Institute. CTCAE version 5.0.

Examples

```
adverseEvent(
  "P001", "Nausea", as.Date("2026-01-02"),
  ctcae_grade = 2, causality = "possible"
)
```

aeSummary

Summarize Adverse Events by Trial Arm

Description

Reports event counts separately from participant incidence, and counts SAE status from the explicit serious field rather than a CTCAE grade cutoff.

Usage

```
aeSummary(events, allocation, arms = NULL)
```

Arguments

- events List of `adverseEvent()` rows or a data frame using the same schema.
- allocation Data frame with unique participant_id and arm.
- arms Optional complete arm order, including zero-event arms.

Value

An ae_summary with arm, severity, and term tables.

Examples

```
allocation <- data.frame(
  participant_id = c("p1", "p2"),
  arm = c("active", "control")
)
events <- list(adverseEvent(
  "p1", "Nausea", as.Date("2026-01-02"),
  ctcae_grade = 1
))
aeSummary(events, allocation, arms = c("active", "control"))
```

allocationRatio	<i>Allocation Ratio</i>
-----------------	-------------------------

Description

Allocation Ratio

Usage

```
allocationRatio(x)

## S4 method for signature 'Trial'
allocationRatio(x)
```

Arguments

- x A `Trial` object.

Value

Named integer vector of allocation weights.

Examples

```
allocationRatio(Trial("T1", c("active", "control")))
```

allocationTable	<i>Allocation Counts</i>
-----------------	--------------------------

Description

Counts use revealed rows only. The stratum value "`__overall__`" denotes totals across all revealed strata.

Usage

```
allocationTable(seq)
```

```
## S4 method for signature 'RandomizationSequence'
allocationTable(seq)
```

Arguments

seq A [RandomizationSequence](#) object.

Value

Data frame with stratum, arm, and n columns.

Examples

```
sequence <- randomize(Trial("T1", c("A", "B")), n = 4, seed = 1)
allocationTable(sequence)
```

analysisData	<i>Apply an Analysis Set and Missing-Data Method</i>
--------------	--

Description

Filters outcome rows to analysis-set members. LOCF is included only for legacy and sensitivity workflows; it is generally unsuitable for primary confirmatory inference. Multiple imputation requires an explicit callback and never silently substitutes a single imputation.

Usage

```
analysisData(
  set,
  outcomes,
  id_col = "id",
  time_col = NULL,
  value_cols = NULL,
  missing = c("none", "locf", "multiple"),
```

```

    imputer = NULL,
    ...
  )

```

Arguments

set	An analysis_set.
outcomes	Outcome data frame.
id_col	Participant-ID column in outcomes.
time_col	Optional time column, required for LOCF.
value_cols	Columns subject to missing-data processing. Defaults to every non-ID/non-time column.
missing	One of "none", "locf", or "multiple".
imputer	Explicit multiple-imputation callback.
...	Arguments forwarded to imputer.

Value

A filtered data frame, or a non-empty list of completed data frames for a multiple-imputation call-back.

Examples

```

trial <- Trial("T1", c("A", "B"))
p <- data.frame(
  id = c("p1", "p2"), arm = c("A", "B"), randomized = TRUE
)
set <- intentionToTreat(trial, p)
outcomes <- data.frame(id = c("p1", "p1", "p2"), time = c(0, 1, 0),
  value = c(NA, 2, 9))
analysisData(set, outcomes, time_col = "time", missing = "locf")

```

analysisSetAccessors *Extract Analysis-Set Rows and Exclusions*

Description

Extract Analysis-Set Rows and Exclusions

Usage

```

analysisParticipants(x)

analysisExclusions(x)

```

Arguments

x An analysis_set.

Value

analysisParticipants() returns the included original rows; analysisExclusions() returns one explicit exclusion row per excluded source participant.

Examples

```
trial <- Trial("T1", c("A", "B"))
p <- data.frame(
  id = c("p1", "screen"),
  arm = c("A", NA),
  randomized = c(TRUE, FALSE)
)
x <- intentionToTreat(trial, p)
analysisParticipants(x)
analysisExclusions(x)
```

Arm	<i>Construct a Trial Arm</i>
-----	------------------------------

Description

Construct a Trial Arm

Usage

```
Arm(
  label,
  ratio = 1L,
  description = NA_character_,
  is_control = FALSE,
  treatment_code = NA_character_
)
```

Arguments

label Non-empty arm label.
ratio Positive integer allocation weight.
description Optional description.
is_control Whether this is a control arm.
treatment_code Optional masked treatment code.

Value

An [Arm](#) object.

Examples

```
Arm("active", ratio = 2L, is_control = FALSE)
```

Arm-class	<i>Trial Arm</i>
-----------	------------------

Description

Trial Arm

Slots

- label Arm label.
- ratio Positive integer allocation weight.
- description Optional description.
- is_control Whether this is a control arm.
- treatment_code Optional masked treatment code.

arms	<i>Arm Labels</i>
------	-------------------

Description

Arm Labels

Usage

```
arms(x)  
  
## S4 method for signature 'Trial'  
arms(x)
```

Arguments

x A [Trial](#) object.

Value

Character vector of arm labels in allocation order.

Examples

```
arms(Trial("T1", c("active", "control")))
```

assignments	<i>Revealed Assignments</i>
-------------	-----------------------------

Description

Revealed Assignments

Usage

```
assignments(seq)

## S4 method for signature 'RandomizationSequence'
assignments(seq)
```

Arguments

seq A [RandomizationSequence](#) object.

Value

Assignment table with every unrevealed arm replaced by NA.

Examples

```
sequence <- randomize(Trial("T1", c("A", "B")), n = 4, seed = 1)
assignments(sequence)
```

auditLog	<i>Randomization Audit Log</i>
----------	--------------------------------

Description

Randomization Audit Log

Usage

```
auditLog(x)

## S4 method for signature 'RandomizationSequence'
auditLog(x)
```

Arguments

x A [RandomizationSequence](#) object.

Value

Append-only audit data frame.

Examples

```
sequence <- randomize(Trial("T1", c("A", "B")), n = 4, seed = 1)
auditLog(sequence)
```

blindingCodes	<i>Masked Treatment Codes</i>
---------------	-------------------------------

Description

Masked Treatment Codes

Usage

```
blindingCodes(manager)
```

Arguments

manager A [BlindingManager](#) object.

Value

Named character vector mapping arms to masked codes.

Examples

```
manager <- blindingManager(Trial("T1", c("A", "B")), seed = 1)
blindingCodes(manager)
```

blindingIndex	<i>Bang Blinding Index</i>
---------------	----------------------------

Description

Computes Bang's arm-specific blinding index and normal-approximation confidence interval. A guess matching the actual arm is correct; dont_know is neutral; any other value, including an unknown arm label or missing guess, is incorrect.

Usage

```
blindingIndex(
  guesses,
  arm_col = "arm",
  guess_col = "guess",
  dont_know = "dont_know",
  conf_level = 0.95
)
```

Arguments

guesses	Data frame containing actual assignments and guesses.
arm_col	Name of the actual-assignment column.
guess_col	Name of the guess column.
dont_know	Sentinel used for a "don't know" response.
conf_level	Confidence level strictly between zero and one.

Value

A `blinding_index` object.

References

Bang H, Ni L, Davis CE (2004). Assessment of blinding in clinical trials. *Controlled Clinical Trials*, 25, 143-156.

Examples

```
guesses <- data.frame(
  arm = c("active", "active", "control", "control"),
  guess = c("active", "dont_know", "active", "control")
)
blindingIndex(guesses)
```

blindingManager

Construct a Blinding Manager

Description

Construct a Blinding Manager

Usage

```
blindingManager(trial, sequence = NULL, codes = NULL, seed = NULL)
```

Arguments

- trial A [Trial](#) object.
- sequence Optional [RandomizationSequence](#) object from the same trial.
- codes Optional named character vector mapping arms to masked treatment codes.
- seed Optional positive integer used to scramble generated codes.

Value

A [BlindingManager](#) object.

Examples

```
trial <- Trial("T1", c("active", "control"))
blindingManager(trial, seed = 1)
```

BlindingManager-class	<i>Blinding and Treatment-Code Manager</i>
-----------------------	--

Description

Blinding and Treatment-Code Manager

Slots

- trial_id Trial identifier.
- arms Ordered arm labels.
- code_map Named arm-to-code mapping.
- kit_codes Concealed participant-level kit assignments.
- unblinding_log Append-only unblinding log.
- fingerprint Deterministic semantic-content hash.

consortCounts	<i>CONSORT Node Counts</i>
---------------	----------------------------

Description

CONSORT Node Counts

Usage

```
consortCounts(x)
```

Arguments

x A consort_diagram.

Value

Node data frame used by the rendered diagram.

Examples

```
# See consortDiagram() for a complete flow example.
```

consortDiagram	<i>CONSORT Participant-Flow Diagram</i>
----------------	---

Description

Builds reconciled CONSORT participant counts and Graphviz DOT. DiagrammeR rendering is optional; the inspectable counts and DOT are always returned.

Usage

```
consortDiagram(
  trial,
  flow,
  render = requireNamespace("DiagrammeR", quietly = TRUE),
  show_reasons = TRUE
)
```

Arguments

trial	A Trial object.
flow	One row per assessed participant using the documented flow schema.
render	Whether to create a <code>DiagrammeR::grViz()</code> widget.
show_reasons	Whether exclusion/disposition reasons appear in labels.

Value

A `consort_diagram` object containing nodes, edges, reason counts, DOT, and an optional widget.

References

Schulz KF, Altman DG, Moher D (2010). CONSORT 2010 Statement.

Examples

```

trial <- Trial("T1", c("active", "control"))
flow <- data.frame(
  participant_id = c("p1", "p2", "p3"),
  eligible = c(TRUE, TRUE, FALSE),
  randomized = c(TRUE, TRUE, FALSE),
  arm = c("active", "control", NA),
  received_allocated = c(TRUE, TRUE, NA),
  followup_complete = c(TRUE, TRUE, NA),
  analysed = c(TRUE, TRUE, NA),
  pre_exclusion_reason = c(NA, NA, "ineligible"),
  not_received_reason = c(NA, NA, NA),
  followup_reason = c(NA, NA, NA),
  analysis_exclusion_reason = c(NA, NA, NA)
)
consortDiagram(trial, flow, render = FALSE)

```

defineXML

*Create a Deterministic define.xml Metadata Stub***Description**

Serializes metadata for the exact datasets in an SDTM-shaped or ADaM-shaped export. The result is a deterministic ODM metadata stub, not a submission-ready define.xml document.

Usage

```

defineXML(
  x,
  file = NULL,
  study_name = NULL,
  protocol_name = NULL,
  standard_version = NULL
)

```

Arguments

x	An sdtm_export or adam_export.
file	Optional output path. Writing uses a temporary file followed by an atomic rename.
study_name, protocol_name	Optional display metadata.
standard_version	Optional standard-version label.

Value

Invisibly, the XML string when file is NULL; otherwise the normalized written path.

Examples

```
trial <- Trial("T1", c("A", "B"))
participants <- data.frame(id = c("p1", "p2"), arm = c("A", "B"))
xml <- defineXML(toSDTM(trial, participants))
grepl("ItemGroupDef", xml, fixed = TRUE)
```

endpointANCOVA

ANCOVA Endpoint Analysis

Description

Fits change-from-baseline or follow-up ANCOVA with treatment referenced to an explicit control. Reported Cohen d values are unadjusted change-score effects delegated to `PhysioExperiment::cohensD()` and are separate from the adjusted ANCOVA estimate.

Usage

```
endpointANCOVA(
  data,
  followup,
  baseline,
  treatment,
  covariates = NULL,
  control = NULL,
  response_scale = c("change", "followup"),
  conf_level = 0.95
)
```

Arguments

<code>data</code>	Participant-level endpoint data.
<code>followup, baseline, treatment</code>	Scalar column names.
<code>covariates</code>	Optional additional model columns.
<code>control</code>	Control-arm label; defaults to the first observed arm.
<code>response_scale</code>	"change" or "followup".
<code>conf_level</code>	Confidence level for adjusted treatment estimates.

Value

An `endpoint_analysis` with standardized estimates, the `lm` fit, complete analysis data, and missingness counts.

Examples

```
d <- data.frame(
  arm = rep(c("control", "active"), each = 4),
  baseline = rep(0:3, 2),
  followup = c(1, 2, 4, 5, 3, 4, 6, 7)
)
endpointANCOVA(d, "followup", "baseline", "arm")
```

endpointMMRM

*MMRM Endpoint Analysis***Description**

Delegates longitudinal modelling and estimated marginal means to `PhysioClinStats`. No alternate MMRM or complete-case ANCOVA fallback is implemented here.

Usage

```
endpointMMRM(
  data,
  response,
  treatment,
  time,
  subject,
  baseline = NULL,
  covariates = NULL,
  control = NULL,
  target_time = NULL,
  covariance = c("unstructured", "ar1", "compound-symmetry", "toeplitz"),
  df = c("kenward-roger", "satterthwaite"),
  conf_level = 0.95
)
```

Arguments

<code>data</code>	Longitudinal endpoint data.
<code>response, treatment, time, subject</code>	Scalar column names.
<code>baseline</code>	Optional repeated subject-level baseline column.
<code>covariates</code>	Optional additional fixed-effect columns.
<code>control</code>	Control-arm label.
<code>target_time</code>	Optional one requested time level.
<code>covariance</code>	MMRM covariance structure.
<code>df</code>	Denominator degrees-of-freedom method.
<code>conf_level</code>	Confidence level.

Value

An endpoint_analysis carrying standardized treatment contrasts and the underlying PhysioClinStats result objects.

`estimateBaselineCorrelation`*Estimate the baseline-outcome correlation (for ANCOVA planning)*

Description

From pilot data - a pair of numeric vectors, a data.frame with a baseline and follow-up column, or a PhysioCohort cohortDesign() plus column names - estimate ρ to feed `sampleSizeANCOVA()`.

Usage

```
estimateBaselineCorrelation(  
  baseline,  
  followup = NULL,  
  cols = NULL,  
  method = "pearson"  
)
```

Arguments

baseline	A numeric vector of baseline values, a data.frame, or a PhysioCohort.
followup	A numeric vector of follow-up values (when baseline is a vector).
cols	Length-2 character vector naming the baseline and follow-up columns (when baseline is a data.frame / cohort design).
method	Correlation method (default "pearson").

Value

The estimated correlation (numeric).

See Also

`sampleSizeANCOVA()`

intentionToTreat	<i>Intention-to-Treat Analysis Set</i>
------------------	--

Description

Includes every randomized participant in the originally assigned arm. Missing outcomes, adherence, and post-randomization events do not alter membership.

Usage

```
intentionToTreat(  
  trial,  
  participants,  
  id_col = "id",  
  arm_col = "arm",  
  randomized_col = "randomized"  
)
```

Arguments

trial	A Trial object.
participants	One row per source participant.
id_col, arm_col, randomized_col	Column names in participants.

Value

An `analysis_set` with included rows, member IDs, and explicit exclusions.

Examples

```
trial <- Trial("T1", c("active", "control"))  
participants <- data.frame(  
  id = c("p1", "p2", "screen"),  
  arm = c("active", "control", NA),  
  randomized = c(TRUE, TRUE, FALSE)  
)  
intentionToTreat(trial, participants)
```

nextAllocation	<i>Reveal the Next Allocation</i>
----------------	-----------------------------------

Description

Reveal the Next Allocation

Usage

```
nextAllocation(seq, requester = NA_character_)

## S4 method for signature 'RandomizationSequence'
nextAllocation(seq, requester = NA_character_)
```

Arguments

seq A [RandomizationSequence](#) object.
 requester Optional audit identity.

Value

A list containing participant_id, arm, and the modified sequence.

Examples

```
sequence <- randomize(Trial("T1", c("A", "B")), n = 4, seed = 1)
nextAllocation(sequence, requester = "site")
```

Participant	<i>Construct a Trial Participant</i>
-------------	--------------------------------------

Description

Construct a Trial Participant

Usage

```
Participant(
  id,
  strata = character(0),
  arm = NA_character_,
  enrolled_at = NA_character_
)
```

Arguments

id	Non-empty participant identifier.
strata	Named character vector mapping prognostic factors to levels.
arm	Allocated arm, or NA before allocation.
enrolled_at	Optional enrollment time representation.

Value

A [Participant](#) object.

Examples

```
Participant("P001", strata = c(site = "north"))
```

Participant-class	<i>Trial Participant</i>
-------------------	--------------------------

Description

Trial Participant

Slots

id	Participant identifier.
strata	Named character vector of prognostic-factor levels.
arm	Allocated arm, or NA before allocation.
enrolled_at	Optional enrollment time representation.

perProtocol	<i>Per-Protocol Analysis Set</i>
-------------	----------------------------------

Description

Starts from ITT and excludes participants with a major protocol deviation or caller-selected nonadherence/incompletion flags. Multiple exclusion reasons are retained in a deterministic order.

Usage

```
perProtocol(  
  trial,  
  participants,  
  deviations = NULL,  
  id_col = "id",  
  arm_col = "arm",  
  randomized_col = "randomized",  
  deviation_id_col = "participant_id",  
  major_col = "major",  
  code_col = "code",  
  adherent_col = NULL,  
  completed_col = NULL  
)
```

Arguments

trial	A Trial object.
participants	One row per source participant.
deviations	Optional protocol-deviation data frame.
id_col, arm_col, randomized_col	Column names in participants.
deviation_id_col, major_col, code_col	Columns in deviations.
adherent_col, completed_col	Optional logical columns in participants.

Value

A PP analysis_set.

Examples

```
trial <- Trial("T1", c("active", "control"))  
participants <- data.frame(  
  id = c("p1", "p2", "screen"),  
  arm = c("active", "control", NA),  
  randomized = c(TRUE, TRUE, FALSE),  
  adherent = c(TRUE, FALSE, NA)  
)  
perProtocol(trial, participants, adherent_col = "adherent")
```

`powerBinary`*Power of a two-arm binary endpoint at a given sample size*

Description

Power of a two-arm binary endpoint at a given sample size

Usage

```
powerBinary(n1, p1, p2, sig_level = 0.05, ratio = 1, sides = 2L)
```

Arguments

`n1` Control-arm sample size.
`p1, p2, sig_level, ratio, sides`
As in [sampleSizeBinary\(\)](#).

Value

The achieved power.

See Also

[sampleSizeBinary\(\)](#)

`powerContinuous`*Power of a two-arm continuous endpoint at a given sample size*

Description

Power of a two-arm continuous endpoint at a given sample size

Usage

```
powerContinuous(n1, delta, sd, sig_level = 0.05, ratio = 1, sides = 2L)
```

Arguments

`n1` Control-arm sample size.
`delta, sd, sig_level, ratio, sides`
As in [sampleSizeContinuous\(\)](#).

Value

The achieved power (numeric in 0-1).

See Also

[sampleSizeContinuous\(\)](#), [powerCurve\(\)](#)

Examples

```
powerContinuous(64, delta = 5, sd = 10)    # ~0.80
```

powerCurve

Power curve over a range of sample sizes or effect sizes

Description

Power curve over a range of sample sizes or effect sizes

Usage

```
powerCurve(
  delta,
  sd,
  n1 = NULL,
  deltas = NULL,
  sig_level = 0.05,
  ratio = 1,
  sides = 2L
)
```

Arguments

delta, sd, sig_level, ratio, sides

As in [sampleSizeContinuous\(\)](#).

n1 A vector of control-arm sizes to evaluate (vary n at fixed effect), or NULL to vary the effect instead.

deltas A vector of effect differences to evaluate at fixed n1 (used when n1 is a single value).

Value

A data.frame with columns n1, delta, power.

See Also

[powerContinuous\(\)](#)

Examples

```
powerCurve(delta = 5, sd = 10, n1 = seq(20, 120, 10))
```

RandomizationSequence-class
<i>Concealed Randomization Sequence</i>

Description

The full allocation table is stored internally. Public accessors mask all rows after the reveal pointer.

Slots

- trial_id Trial identifier.
- method Allocation method.
- arms Ordered arm labels.
- ratio Named allocation weights.
- seed Captured random seed.
- strata Trial stratification definition.
- block_sizes Admissible block lengths.
- p_bias Minimization biased-coin probability.
- weights Named minimization factor weights.
- imbalance Minimization discrepancy measure.
- table Sealed allocation table.
- audit Append-only access audit.
- fingerprint Deterministic semantic-content hash.
- revealed Number of revealed rows.

randomize	<i>Generate a Concealed Randomization Sequence</i>
-----------	--

Description

Generate a Concealed Randomization Sequence

Usage

```
randomize(  
  trial,  
  method = c("simple", "permuted_block", "stratified_block", "minimization"),  
  n = NULL,  
  participants = NULL,  
  block_sizes = NULL,  
  seed = NULL,
```

```
p_bias = 0.8,
weights = NULL,
imbalance = c("range", "variance", "sd"),
requester = NA_character_
)
```

Arguments

trial	A Trial object.
method	Allocation method. Simple randomization matches the allocation ratio only in expectation. Block methods enforce the ratio in every completed block.
n	Number of anonymous slots. Valid only for an unstratified trial and the simple or permuted-block methods.
participants	A data frame with id and trial-factor columns, or a list of Participant objects.
block_sizes	Positive block lengths, each a multiple of the allocation weight sum.
seed	Positive integer seed. If NULL, a seed is drawn and captured.
p_bias	High-probability branch of minimization's biased coin.
weights	Optional named prognostic-factor weights.
imbalance	Pocock-Simon discrepancy measure.
requester	Optional audit identity recorded at sealing.

Value

A concealed [RandomizationSequence](#) object.

Examples

```
trial <- Trial("T1", c("active", "control"))
randomize(trial, "permuted_block", n = 8, seed = 2026)
```

randomSeed	<i>Captured Randomization Seed</i>
------------	------------------------------------

Description

Captured Randomization Seed

Usage

```
randomSeed(seq)

## S4 method for signature 'RandomizationSequence'
randomSeed(seq)
```

Arguments

seq A [RandomizationSequence](#) object.

Value

Positive integer seed.

Examples

```
sequence <- randomize(Trial("T1", c("A", "B")), n = 4, seed = 19)
randomSeed(sequence)
```

responderAnalysis	<i>MDC/MCID Responder Analysis</i>
-------------------	------------------------------------

Description

Delegates each subject's dual-threshold classification to [PhysioClinical::classifyResponder\(\)](#) and summarizes true responders with Wilson score intervals. Missing outcomes remain explicit and are excluded from the rate denominator.

Usage

```
responderAnalysis(
  data,
  baseline,
  followup,
  treatment,
  mdc,
  mcid,
  direction = c("increase", "decrease"),
  control = NULL,
  conf_level = 0.95
)
```

Arguments

data	Participant-level data.
baseline, followup, treatment	Scalar column names.
mdc, mcid	Positive MDC and MCID thresholds.
direction	Whether improvement is an increase or decrease.
control	Optional control arm placed first.
conf_level	Wilson interval confidence level.

Value

A responder_analysis with subject classifications and arm summaries.

sampleSizeANCOVA

*Sample size for a baseline-adjusted (ANCOVA) continuous endpoint***Description**

The recommended pre-post rehabilitation analysis: adjusting the follow-up outcome for its baseline reduces the residual variance by a factor $(1 - \rho^2)$, where ρ is the baseline-outcome correlation - shrinking the required sample versus an unadjusted comparison.

Usage

```
sampleSizeANCOVA(
  delta,
  sd,
  baseline_cor,
  power = 0.8,
  sig_level = 0.05,
  ratio = 1,
  sides = 2L,
  dropout = 0,
  cluster_size = NULL,
  icc = NULL
)
```

Arguments

delta, sd, power, sig_level, ratio, sides, dropout, cluster_size, icc
 As in [sampleSizeContinuous\(\)](#). sd is the raw (unadjusted) outcome SD.

baseline_cor Correlation ρ between baseline and follow-up outcome (0-1); the variance is scaled by $(1 - \rho^2)$.

Value

A trial_power object.

References

Frison & Pocock (1992), Stat Med; Borm et al. (2007), J Clin Epi.

See Also

[sampleSizeChangeScore\(\)](#), [estimateBaselineCorrelation\(\)](#)

Examples

```
sampleSizeANCOVA(delta = 5, sd = 10, baseline_cor = 0.6)    # < 64/arm
```

sampleSizeBinary	<i>Sample size for a two-arm binary / responder endpoint</i>
------------------	--

Description

Comparison of two proportions (e.g. responder rates), via the normal approximation with unequal allocation; matches `stats::power.prop.test` in the balanced case.

Usage

```
sampleSizeBinary(  
  p1,  
  p2,  
  power = 0.8,  
  sig_level = 0.05,  
  ratio = 1,  
  sides = 2L,  
  dropout = 0,  
  cluster_size = NULL,  
  icc = NULL  
)
```

Arguments

`p1` Control-arm proportion.
`p2` Treatment-arm proportion.
`power`, `sig_level`, `ratio`, `sides`, `dropout`, `cluster_size`, `icc`
As in [sampleSizeContinuous\(\)](#).

Value

A `trial_power` object.

See Also

[sampleSizeContinuous\(\)](#), [powerBinary\(\)](#)

Examples

```
sampleSizeBinary(p1 = 0.30, p2 = 0.50)    # responder-rate 30% vs 50%
```

sampleSizeChangeScore *Sample size for a change-from-baseline continuous endpoint*

Description

Analysis of the change score (follow-up minus baseline). The change SD is $\sigma\sqrt{2(1-\rho)}$; change-score analysis beats an unadjusted follow-up comparison only when $\rho > 0.5$, and is never better than ANCOVA.

Usage

```
sampleSizeChangeScore(
  delta,
  sd,
  baseline_cor,
  power = 0.8,
  sig_level = 0.05,
  ratio = 1,
  sides = 2L,
  dropout = 0,
  cluster_size = NULL,
  icc = NULL
)
```

Arguments

delta, sd, baseline_cor, power, sig_level, ratio, sides, dropout,
cluster_size, icc

As in [sampleSizeANCOVA\(\)](#).

Value

A trial_power object.

See Also

[sampleSizeANCOVA\(\)](#)

sampleSizeContinuous *Sample size for a two-arm continuous endpoint*

Description

Two-sample comparison of means. Delegates the balanced case's core to the noncentral-t (matching `stats::power.t.test`) and adds unequal allocation, cluster design effect, and dropout inflation.

Usage

```
sampleSizeContinuous(
  delta,
  sd,
  power = 0.8,
  sig_level = 0.05,
  ratio = 1,
  sides = 2L,
  dropout = 0,
  cluster_size = NULL,
  icc = NULL
)
```

Arguments

<code>delta</code>	Target difference in means (treatment - control).
<code>sd</code>	Common (residual) standard deviation of the endpoint.
<code>power</code>	Target power (default 0.8).
<code>sig_level</code>	Two-sided (or one-sided, see <code>sides</code>) alpha (default 0.05).
<code>ratio</code>	Allocation ratio n_2/n_1 (treatment/control; default 1).
<code>sides</code>	1 or 2 (default 2).
<code>dropout</code>	Expected proportion lost to follow-up; inflates enrolment (default 0).
<code>cluster_size</code>	Average subjects per cluster for a cluster-randomized design (NULL = individually randomized).
<code>icc</code>	Intra-cluster correlation (required with <code>cluster_size</code>).

Value

A `trial_power` object (`n1`, `n2`, `n_total`, `n_enrolled`, design effect, clusters, ...).

See Also

[sampleSizeANCOVA\(\)](#), [sampleSizeBinary\(\)](#), [powerContinuous\(\)](#)

Examples

```
sampleSizeContinuous(delta = 5, sd = 10)           # d = 0.5 -> 64/arm
sampleSizeContinuous(delta = 5, sd = 10, dropout = 0.2)
```

sampleSizeRepeatedMeasures

Sample size for a repeated-measures continuous endpoint

Description

Comparison of the mean of `n_measurements` post-baseline measurements under a compound-symmetry correlation correlation between them. The effective variance is $\sigma^2[1 + (r - 1)\rho]/r$, so more visits help most when the within-subject correlation is low.

Usage

```
sampleSizeRepeatedMeasures(
  delta,
  sd,
  n_measurements,
  correlation,
  power = 0.8,
  sig_level = 0.05,
  ratio = 1,
  sides = 2L,
  dropout = 0,
  cluster_size = NULL,
  icc = NULL
)
```

Arguments

`delta`, `sd`, `power`, `sig_level`, `ratio`, `sides`, `dropout`, `cluster_size`, `icc`

As in [sampleSizeContinuous\(\)](#).

`n_measurements` Number of post-baseline measurements per subject.

`correlation` Compound-symmetry correlation between measurements (0-1).

Value

A `trial_power` object.

References

Frison & Pocock (1992), Stat Med.

See Also

[sampleSizeANCOVA\(\)](#)

spiritChecklist	<i>SPIRIT 2013 Protocol Checklist</i>
-----------------	---------------------------------------

Description

Marks protocol evidence against the bundled SPIRIT 2013 checklist. This is a completeness aid and does not certify protocol compliance.

Usage

```
spiritChecklist(protocol, checklist = NULL)
```

Arguments

protocol	Named list (item_id to evidence) or data frame with item_id and evidence.
checklist	Optional replacement checklist using the bundled schema.

Value

A spirit_checklist data frame with completion flags and evidence.

References

Chan AW et al. (2013). SPIRIT 2013 Statement.

Examples

```
spiritChecklist(list(`1` = "Descriptive title", `2` = "Registry ID"))
```

spiritSchedule	<i>SPIRIT Schedule of Events</i>
----------------	----------------------------------

Description

Converts a validated long schedule into a deterministic wide enrolment/intervention/assessment table.

Usage

```
spiritSchedule(trial, events, marker = "X")
```

Arguments

trial	A Trial object.
events	Long schedule data frame using the documented event schema.
marker	Default marker for scheduled cells.

Value

A spirit_schedule object with validated long events, a wide table, and ordered timepoint meta-data.

References

Chan AW et al. (2013). SPIRIT 2013 Statement.

Examples

```
trial <- Trial("T1", c("active", "control"))
events <- data.frame(
  timepoint_id = c("screen", "week12"),
  timepoint_label = c("Screening", "Week 12"),
  timepoint_order = c(-1L, 1L),
  category = c("enrolment", "assessment"),
  activity = c("Eligibility", "Primary endpoint"),
  arm = NA_character_,
  scheduled = TRUE
)
spiritSchedule(trial, events)
```

strata	<i>Trial Strata</i>
--------	---------------------

Description

Trial Strata

Usage

```
strata(x)

## S4 method for signature 'Trial'
strata(x)
```

Arguments

x A [Trial](#) object.

Value

Named list of prognostic factors and admissible levels.

Examples

```
strata(Trial(
  "T1", c("active", "control"),
  strata = list(site = c("north", "south"))
))
```

toADaM

*Export Trial Analysis Sets to ADaM-Shaped Data***Description**

Builds an ADSL population table and a generic BDS analysis table from explicit analysis sets. This structural export is not a claim of formal ADaM conformance.

Usage

```
toADaM(
  set,
  subject_data,
  bds = NULL,
  study_id = set$trial_id,
  id_col = "id",
  treatment_col = "arm",
  population_flags = NULL
)
```

Arguments

<code>set</code>	The authoritative primary analysis_set.
<code>subject_data</code>	One row per source participant.
<code>bds</code>	Optional generic BDS source data frame. A named list may contain an ADBDS source plus already constructed analysis datasets such as ADQS.
<code>study_id</code>	Non-empty study identifier.
<code>id_col, treatment_col</code>	Subject ID and originally assigned treatment columns.
<code>population_flags</code>	Optional named list of other analysis_set objects from the same source population.

Value

An adam_export containing datasets, manifest, and population metadata.

Examples

```
trial <- Trial("T1", c("active", "control"))
subjects <- data.frame(
  id = c("p1", "p2", "screen"),
  arm = c("active", "control", NA),
  randomized = c(TRUE, TRUE, FALSE)
)
itt <- intentionToTreat(trial, subjects)
export <- toADaM(itt, subjects)
export$datasets$ADSL
```

toSDTM

*Export Trial Data to CDISC SDTM-Shaped Domains***Description**

Builds deterministic, linked trial domains with explicit sponsor mappings. The output is structurally SDTM-shaped; it is not a claim of CDISC, regulator, or third-party validator conformance.

Usage

```
toSDTM(
  trial,
  participants,
  adverse_events = NULL,
  findings = list(),
  study_id = trial@id,
  id_col = "id",
  arm_col = "arm",
  sex_col = NULL,
  birth_date_col = NULL,
  reference_start_col = NULL,
  reference_end_col = NULL,
  country_col = NULL,
  site_col = NULL,
  extra_domains = list()
)
```

Arguments

<code>trial</code>	A Trial object.
<code>participants</code>	One row per screened or enrolled participant.
<code>adverse_events</code>	Optional adverse events accepted by aeSummary() .
<code>findings</code>	Named list containing normalized VS, EG, or XP findings data frames.
<code>study_id</code>	Non-empty study identifier.
<code>id_col, arm_col</code>	Participant ID and assigned-arm columns.
<code>sex_col, birth_date_col, reference_start_col, reference_end_col</code>	Optional participant columns.
<code>country_col, site_col</code>	Optional country and site columns.
<code>extra_domains</code>	Named list of already constructed two-letter domains, such as output from <code>PhysioClinical::toCDISC_Q</code>

Value

An `sdtm_export` containing datasets, manifest, and mapping metadata.

Examples

```
trial <- Trial("T1", c("active", "control"))
participants <- data.frame(
  id = c("p1", "p2"),
  arm = c("active", "control")
)
export <- toSDTM(trial, participants)
export$datasets$DM
```

Trial

*Construct a Randomized Trial***Description**

Construct a Randomized Trial

Usage

```
Trial(
  id,
  arms,
  allocation_ratio = NULL,
  strata = list(),
  endpoints = list(),
  metadata = list()
)
```

Arguments

id	Non-empty trial identifier.
arms	Character arm labels or a list of Arm objects.
allocation_ratio	Optional named or positional allocation weights.
strata	Named list mapping prognostic factors to admissible levels.
endpoints	Named endpoint definitions.
metadata	Free-form trial metadata.

Value

A [Trial](#) object.

Examples

```
Trial(
  "rehab-01",
  arms = c("active", "control"),
  strata = list(site = c("north", "south"))
)
```

Trial-class	<i>Randomized Trial Definition</i>
-------------	------------------------------------

Description

Randomized Trial Definition

Slots

id Trial identifier.
 arms Ordered arm labels.
 arm_specs Arm definitions.
 allocation_ratio Named integer allocation weights.
 strata Named list of prognostic factors and admissible levels.
 endpoints Endpoint definitions carried by the trial.
 metadata Free-form trial metadata.

trialFingerprint	<i>Trial Object Fingerprint</i>
------------------	---------------------------------

Description

Trial Object Fingerprint

Usage

```
trialFingerprint(x)

## S4 method for signature 'RandomizationSequence'
trialFingerprint(x)

## S4 method for signature 'BlindingManager'
trialFingerprint(x)
```

Arguments

x A [RandomizationSequence](#) or [BlindingManager](#) object.

Value

Deterministic xxhash64 semantic-content fingerprint.

Examples

```
sequence <- randomize(Trial("T1", c("A", "B")), n = 4, seed = 1)
trialFingerprint(sequence)
```

unblind	<i>Record an Unblinding Event</i>
---------	-----------------------------------

Description

Record an Unblinding Event

Usage

```
unblind(  
  manager,  
  participant_id,  
  requester = NA_character_,  
  reason = NA_character_  
)
```

Arguments

manager	A BlindingManager object.
participant_id	Participant to unblind.
requester	Optional identity requesting unblinding.
reason	Optional reason for unblinding.

Value

The modified [BlindingManager](#) object.

Examples

```
trial <- Trial("T1", c("A", "B"))  
sequence <- randomize(trial, n = 4, seed = 1)  
manager <- blindingManager(trial, sequence, seed = 2)  
unblind(manager, "slot_1", requester = "PI", reason = "safety")
```

unblindingLog	<i>Unblinding Log</i>
---------------	-----------------------

Description

Unblinding Log

Usage

```
unblindingLog(manager)
```

Arguments

manager A [BlindingManager](#) object.

Value

Append-only unblinding data frame.

Examples

```
manager <- blindingManager(Trial("T1", c("A", "B")), seed = 1)
unblindingLog(manager)
```

 validateCDISC

Validate a CDISC-Shaped Export

Description

Checks the documented structural contract for exports created by [toSDTM\(\)](#) or [toADaM\(\)](#). A clean report does not mean an external conformance product or regulator has accepted the data.

Usage

```
validateCDISC(x, controlled_terms = NULL)
```

Arguments

x An sdtm_export or adam_export.

controlled_terms Optional named list, or a data frame with variable and value, adding caller-controlled terminology checks.

Value

A cdisc_validation object with deterministic issues and summary.

Examples

```
trial <- Trial("T1", c("A", "B"))
participants <- data.frame(id = c("p1", "p2"), arm = c("A", "B"))
validateCDISC(toSDTM(trial, participants))
```

Index

adverseEvent, [3](#)
adverseEvent(), [5](#)
aeSummary, [4](#)
aeSummary(), [36](#)
allocationRatio, [5](#)
allocationRatio, Trial-method
 (allocationRatio), [5](#)
allocationTable, [6](#)
allocationTable, RandomizationSequence-method
 (allocationTable), [6](#)
analysisData, [6](#)
analysisExclusions
 (analysisSetAccessors), [7](#)
analysisParticipants
 (analysisSetAccessors), [7](#)
analysisSetAccessors, [7](#)
Arm, [8](#), [8](#), [37](#)
Arm-class, [9](#)
arms, [9](#)
arms, Trial-method (arms), [9](#)
assignments, [10](#)
assignments, RandomizationSequence-method
 (assignments), [10](#)
auditLog, [10](#)
auditLog, RandomizationSequence-method
 (auditLog), [10](#)

blindingCodes, [11](#)
blindingIndex, [11](#)
BlindingManager, [11](#), [13](#), [38–40](#)
blindingManager, [12](#)
BlindingManager-class, [13](#)

consortCounts, [13](#)
consortDiagram, [14](#)

defineXML, [15](#)

endpointANCOVA, [16](#)
endpointMMRM, [17](#)

estimateBaselineCorrelation, [18](#)
estimateBaselineCorrelation(), [28](#)

intentionToTreat, [19](#)

nextAllocation, [20](#)
nextAllocation, RandomizationSequence-method
 (nextAllocation), [20](#)

Participant, [20](#), [21](#), [26](#)
Participant-class, [21](#)
perProtocol, [21](#)
PhysioClinical::classifyResponder(),
 [27](#)
PhysioExperiment::cohensD(), [16](#)
powerBinary, [23](#)
powerBinary(), [29](#)
powerContinuous, [23](#)
powerContinuous(), [24](#), [31](#)
powerCurve, [24](#)
powerCurve(), [24](#)

RandomizationSequence, [6](#), [10](#), [13](#), [20](#), [26](#),
 [27](#), [38](#)
RandomizationSequence-class, [25](#)
randomize, [25](#)
randomSeed, [26](#)
randomSeed, RandomizationSequence-method
 (randomSeed), [26](#)
responderAnalysis, [27](#)

sampleSizeANCOVA, [28](#)
sampleSizeANCOVA(), [18](#), [30–32](#)
sampleSizeBinary, [29](#)
sampleSizeBinary(), [23](#), [31](#)
sampleSizeChangeScore, [30](#)
sampleSizeChangeScore(), [28](#)
sampleSizeContinuous, [30](#)
sampleSizeContinuous(), [23](#), [24](#), [28](#), [29](#), [32](#)
sampleSizeRepeatedMeasures, [32](#)
spiritChecklist, [33](#)

- spiritSchedule, [33](#)
- strata, [34](#)
- strata, Trial-method (strata), [34](#)

- toADaM, [35](#)
- toADaM(), [40](#)
- toSDTM, [36](#)
- toSDTM(), [40](#)
- Trial, [5](#), [9](#), [13](#), [14](#), [19](#), [22](#), [26](#), [33](#), [34](#), [36](#), [37](#),
[37](#)
- Trial-class, [38](#)
- trialFingerprint, [38](#)
- trialFingerprint, BlindingManager-method
(trialFingerprint), [38](#)
- trialFingerprint, RandomizationSequence-method
(trialFingerprint), [38](#)

- unblind, [39](#)
- unblindingLog, [39](#)

- validateCDISC, [40](#)