

Package: cmrdesign (via r-universe)

July 24, 2026

Title Conditional Minimax Regret Design Rules

Version 0.0.0.9000

Author Juan Yamin Silva [aut, cre]

Maintainer Juan Yamin Silva <juan_yamin_silva@brown.edu>

Description Implements Conditional Minimax Regret design rules for pilot-informed experimental assignment, including two-arm, unbounded two-arm, multi-arm, stratified, multiple-outcome, proxy-outcome, and pilot-planning tools.

License MIT + file LICENSE

URL <https://github.com/juancyamin/cmrdesign>,
<https://arxiv.org/abs/2607.16982>

BugReports <https://github.com/juancyamin/cmrdesign/issues>

Depends R (>= 4.1.0)

Encoding UTF-8

Language en-US

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

VignetteBuilder knitr

Imports stats

Suggests testthat (>= 3.0.0), jsonlite, knitr, rmarkdown

Config/testthat/edition 3

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

Date/Publication 2026-07-24 01:02:17 UTC

RemoteUrl <https://github.com/juancyamin/cmrdesign>

RemoteRef HEAD

RemoteSha 1ec3aee4d415b1e0a831765d3ba3efb890f6ed6d

RemoteSubdir r

Contents

cmrdesign-package	2
activation_threshold_bounded	3
assign_balance	4
binary_rectangle_corners	5
break_even_pilot_share	6
cmr_multiarm	7
cmr_multiarm_from_rectangle	8
cmr_multiple_outcomes	9
cmr_proxy	11
cmr_stratified	12
cmr_stratified_from_rectangle	14
cmr_two_arm	15
cmr_unbounded	17
cmr_unbounded_from_rectangle	18
coverage_indicator	19
folded_binomial_pmf	20
multiarm_variance_objective	21
pilot_plan	22
pilot_viability_band	23
print.cmr_two_arm	24
rectangle_bernoulli_binary	26
rectangle_bounded_two_arm	28
rectangle_multiarm	30
rectangle_multiple_outcomes	31
rectangle_proxy	32
rectangle_stratified	34
rectangle_unbounded	36
stratified_variance_objective	37
variance_bounds_bernoulli_exact	38
variance_bounds_maurer_pontil	39
variance_bounds_unbounded_mom	40
variance_objective	41
Index	43

cmrdesign-package

cmrdesign: Conditional Minimax Regret design rules

Description

Applied Conditional Minimax Regret (CMR) design rules for pilot-informed experiments. The package includes two-arm, shared-control multi-arm, stratified, multiple-outcome, proxy/delayed-outcome, unbounded-outcome, and pilot-planning workflows.

Author(s)

Maintainer: Juan Yamin Silva <juan_yamin_silva@brown.edu>

See Also

Useful links:

- <https://github.com/juancyamin/cmrdesign>
- <https://arxiv.org/abs/2607.16982>
- Report bugs at <https://github.com/juancyamin/cmrdesign/issues>

activation_threshold_bounded

Pilot-size activation thresholds

Description

Compute simple method-specific pilot-size activation thresholds for the Appendix E-style pilot-planning screen.

Usage

```
activation_threshold_bounded(  
  alpha = 0.05,  
  max_total_pilot = 10000L,  
  min_arm_size = 2L  
)  
  
activation_threshold_bernoulli(alpha = 0.05)
```

Arguments

alpha	Target error level.
max_total_pilot	Largest total pilot size to search.
min_arm_size	Minimum pilot observations per arm.

Value

Total pilot size threshold. `activation_threshold_bounded()` returns Inf if no even pilot size up to `max_total_pilot` clears the bounded-outcome activation condition. `activation_threshold_bernoulli()` returns 4.

See Also

Other pilot planning: [break_even_pilot_share\(\)](#), [pilot_plan\(\)](#), [pilot_viability_band\(\)](#)

Examples

```
activation_threshold_bounded(alpha = 0.05, max_total_pilot = 200)
activation_threshold_bernoulli(alpha = 0.05)
```

assign_balance	<i>Baseline and comparator assignment rules</i>
----------------	---

Description

Standalone benchmark assignment rules for comparing CMR against balance, feasible Neyman, and simple regularized Neyman variants.

Usage

```
assign_balance(n = 1L)

assign_multiarm_balance(arms)

assign_stratified_balance(strata_share)

assign_feasible_neyman(vhat1, vhat0)

assign_trimmed_neyman(vhat1, vhat0, trim = 0.1)

assign_additive_regularized_neyman(vhat1, vhat0, nu)

assign_exponential_regularized_neyman(
  vhat1,
  vhat0,
  tau,
  zero_guard = c("any", "both", "none")
)
```

Arguments

n	Number of assignment shares to return for assign_balance().
arms	Either the number of treatment arms, excluding control, or a vector of arm labels that includes control arm "0".
strata_share	Named stratum population shares that sum to one.
vhat1	Estimated treatment-arm variance or vector of estimates.
vhat0	Estimated control-arm variance or vector of estimates.
trim	Lower and upper trimming amount for assign_trimmed_neyman(). The returned share is clipped to [trim, 1 - trim].
nu	Nonnegative additive regularization strength.

tau	Nonnegative exponent for exponential regularization.
zero_guard	How zero variance estimates are guarded in <code>assign_exponential_regularized_neyman()</code> : "any" returns balance if either arm variance is zero, "both" only if both are zero, and "none" applies no extra guard.

Value

Numeric assignment shares. Two-arm functions return treatment shares. `assign_multiarm_balance()` returns a named vector over all arms, including control "0". `assign_stratified_balance()` returns total assignment shares for treatment and control cells named like "1:A" and "0:A".

See Also

Other assignment helpers: [multiarm_variance_objective\(\)](#), [stratified_variance_objective\(\)](#), [variance_objective\(\)](#)

Examples

```
assign_balance(3)
assign_feasible_neyman(0.12, 0.04)
assign_trimmed_neyman(0.12, 0.04, trim = 0.10)
assign_multiarm_balance(2)
assign_stratified_balance(c(A = 0.4, B = 0.6))
```

binary_rectangle_corners

Two-arm CMR from a variance rectangle

Description

Compute the closed-form two-arm CMR allocation for a supplied confidence rectangle over treatment and control variances.

Usage

```
binary_rectangle_corners(rectangle)

binary_rectangle_regret(pi, rectangle, return_details = FALSE)

cmr_two_arm_from_rectangle(rectangle)

cmr_binary_from_rectangle(rectangle)
```

Arguments

rectangle	Two-arm variance rectangle, either a numeric vector with names v_l1, v_u1, v_l0, v_u0 or a compatible rectangle object.
pi	Treatment assignment share.
return_details	If TRUE, return corner regrets and binding-corner information instead of only the maximum regret.

Value

cmr_two_arm_from_rectangle() returns a list of class cmr_two_arm with pi, U_CMR, the input rectangle, rectangle corners, corner regrets, binding-corner diagnostics, and additional solver diagnostics. binary_rectangle_corners() returns the two least-favorable rectangle corners. binary_rectangle_regret() returns the worst regret at pi, or a detailed list when return_details = TRUE.

See Also

Other CMR rules: [cmr_multiarm\(\)](#), [cmr_multiarm_from_rectangle\(\)](#), [cmr_multiple_outcomes\(\)](#), [cmr_proxy\(\)](#), [cmr_stratified\(\)](#), [cmr_stratified_from_rectangle\(\)](#), [cmr_two_arm\(\)](#), [cmr_unbounded\(\)](#), [cmr_unbounded_from_rectangle\(\)](#), [print.cmr_two_arm\(\)](#)

Other rectangle helpers: [cmr_multiarm_from_rectangle\(\)](#), [cmr_stratified_from_rectangle\(\)](#), [cmr_unbounded_from_rectangle\(\)](#), [folded_binomial_pmf\(\)](#), [multiarm_variance_objective\(\)](#), [rectangle_bernoulli_binary\(\)](#), [rectangle_bounded_two_arm\(\)](#), [rectangle_multiarm\(\)](#), [rectangle_multiple_outcomes\(\)](#), [rectangle_proxy\(\)](#), [rectangle_stratified\(\)](#), [rectangle_unbounded\(\)](#), [stratified_variance_objective\(\)](#), [variance_bounds_bernoulli_exact\(\)](#), [variance_bounds_maurer_pontil\(\)](#), [variance_bounds_unbounded_mom\(\)](#)

Examples

```
rect <- c(v_l1 = 0.02, v_u1 = 0.12, v_l0 = 0.01, v_u0 = 0.08)
fit <- cmr_two_arm_from_rectangle(rect)
fit$pi
fit$U_CMR
binary_rectangle_regret(0.5, rect)
```

break_even_pilot_share

Break-even pilot share

Description

Compute the design-only break-even pilot share implied by treatment and control standard deviations.

Usage

```
break_even_pilot_share(sigma1, sigma0, input = c("sd", "variance"))
```

Arguments

sigma1	Treatment-arm standard deviation, or variance when input = "variance".
sigma0	Control-arm standard deviation, or variance when input = "variance".
input	Whether sigma1 and sigma0 are standard deviations ("sd") or variances ("variance").

Value

Numeric break-even share in $[0, 0.5]$.

See Also

Other pilot planning: [activation_threshold_bounded\(\)](#), [pilot_plan\(\)](#), [pilot_viability_band\(\)](#)

Examples

```
break_even_pilot_share(sigma1 = 0.35, sigma0 = 0.20)
break_even_pilot_share(sigma1 = 0.35^2, sigma0 = 0.20^2, input = "variance")
```

cmr_multiarm

Shared-control multi-arm CMR assignment

Description

Estimate arm-specific variance confidence intervals from pilot data and return the shared-control multi-arm CMR assignment.

Usage

```
cmr_multiarm(
  y,
  arm,
  alpha = 0.05,
  method = c("auto", "bounded", "bernoulli", "maurer_pontil", "mp", "bernoulli_exact",
    "martinez_taboada_ramdas", "mtr"),
  beta = NULL,
  control_arm = 0,
  normalize = FALSE,
  lower = NULL,
  upper = NULL,
  na.rm = TRUE,
  tol = 1e-11,
  solver_control = list(),
  max_vertices = 65536L
)
```

Arguments

y	Pilot outcomes.
arm	Pilot arm labels. The control arm is identified by control_arm and internally standardized to "0".
alpha	Target joint error level.
method	Confidence-set method. "auto" chooses exact Bernoulli bounds for 0/1 outcomes and bounded Maurer-Pontil bounds otherwise.
beta	Optional endpoint error allocation. If NULL, Bonferroni error is split across all lower and upper arm endpoints.
control_arm	Label identifying the control arm in arm.
normalize	If TRUE, normalize bounded outcomes to $[0, 1]$ before computing variances.
lower, upper	Optional lower and upper outcome bounds used when normalize = TRUE.
na.rm	If TRUE, drop rows with missing y or arm.
tol	Numerical tolerance for exact Bernoulli bound inversion.
solver_control	Optional list of solver controls for the general vertex epigraph solver.
max_vertices	Maximum number of hyperrectangle vertices to enumerate.

Value

A list of class `cmr_multiarm` with named assignment shares `pi` over all arms, CMR certificate `U_CMR`, confidence set, pilot summaries, endpoint error allocation, and diagnostics.

See Also

Other CMR rules: `binary_rectangle_corners()`, `cmr_multiarm_from_rectangle()`, `cmr_multiple_outcomes()`, `cmr_proxy()`, `cmr_stratified()`, `cmr_stratified_from_rectangle()`, `cmr_two_arm()`, `cmr_unbounded()`, `cmr_unbounded_from_rectangle()`, `print.cmr_two_arm()`

Examples

```
set.seed(5)
arm <- rep(c(0, 1, 2), each = 20)
y <- c(rbeta(20, 4, 4), rbeta(20, 2, 6), rbeta(20, 5, 3))
cmr_multiarm(y, arm, method = "bounded")
```

`cmr_multiarm_from_rectangle`

Multi-arm CMR from a variance rectangle

Description

Compute a shared-control multi-arm CMR allocation from a supplied variance rectangle.

Usage

```
cmr_multiarm_from_rectangle(rectangle, control = list(), max_vertices = 65536L)
```

Arguments

rectangle	Multi-arm variance rectangle, either a matrix/data frame with lower and upper columns or a named vector with entries like <code>v_l0</code> , <code>v_u0</code> , <code>v_l1</code> , <code>v_u1</code> .
control	Optional list of solver controls for the general vertex epigraph solver.
max_vertices	Maximum number of hyperrectangle vertices to enumerate.

Value

A list of class `cmr_multiarm` with named assignment shares `pi`, regret certificate `U_CMR`, checked rectangle, enumerated vertices, vertex regrets, binding vertices, and solver diagnostics. For collapsed or full rectangles, closed-form shortcuts are used.

See Also

Other CMR rules: `binary_rectangle_corners()`, `cmr_multiarm()`, `cmr_multiple_outcomes()`, `cmr_proxy()`, `cmr_stratified()`, `cmr_stratified_from_rectangle()`, `cmr_two_arm()`, `cmr_unbounded()`, `cmr_unbounded_from_rectangle()`, `print.cmr_two_arm()`

Other rectangle helpers: `binary_rectangle_corners()`, `cmr_stratified_from_rectangle()`, `cmr_unbounded_from_rectangle()`, `folded_binomial_pmf()`, `multiarm_variance_objective()`, `rectangle_bernoulli_binary()`, `rectangle_bounded_two_arm()`, `rectangle_multiarm()`, `rectangle_multiple_outcomes()`, `rectangle_proxy()`, `rectangle_stratified()`, `rectangle_unbounded()`, `stratified_variance_objective()`, `variance_bounds_bernoulli_exact()`, `variance_bounds_maurer_pontil()`, `variance_bounds_unbounded_mom()`

Examples

```
rect <- c(
  v_l0 = 0.02, v_u0 = 0.08,
  v_l1 = 0.04, v_u1 = 0.12,
  v_l2 = 0.01, v_u2 = 0.07
)
cmr_multiarm_from_rectangle(rect)
```

`cmr_multiple_outcomes` *Multiple-outcome CMR assignment*

Description

Estimate an effective two-arm variance rectangle for multiple outcomes and return the CMR treatment share.

Usage

```
cmr_multiple_outcomes(
  y,
  d,
  weights = NULL,
  estimand = c("coprimary", "index"),
  alpha = 0.05,
  method = c("auto", "bounded", "bernoulli", "maurer_pontil", "mp", "bernoulli_exact",
    "martinez_taboada_ramdas", "mtr"),
  beta = NULL,
  na.rm = TRUE,
  tol = 1e-11
)
```

Arguments

y	Pilot outcomes as a numeric matrix, data frame, or vector. Rows are units and columns are outcomes.
d	Pilot treatment indicator; treatment is 1 and control is 0.
weights	Optional nonnegative outcome weights. If NULL, equal weights are used.
estimand	Either "coprimary" or "index".
alpha	Target joint error level.
method	Confidence-set method. "auto" chooses exact Bernoulli bounds for 0/1 outcomes and bounded Maurer-Pontil bounds otherwise.
beta	Optional scalar endpoint error allocation.
na.rm	If TRUE, drop rows with missing y or d.
tol	Numerical tolerance for exact Bernoulli bound inversion.

Value

A list of class `cmr_multiple_outcomes` and `cmr_two_arm` with treatment share π , CMR certificate U_{CMR} , effective confidence rectangle, pilot summaries, outcome weights, estimand metadata, endpoint error allocation, and diagnostics.

See Also

Other CMR rules: [binary_rectangle_corners\(\)](#), [cmr_multiarm\(\)](#), [cmr_multiarm_from_rectangle\(\)](#), [cmr_proxy\(\)](#), [cmr_stratified\(\)](#), [cmr_stratified_from_rectangle\(\)](#), [cmr_two_arm\(\)](#), [cmr_unbounded\(\)](#), [cmr_unbounded_from_rectangle\(\)](#), [print.cmr_two_arm\(\)](#)

Examples

```
set.seed(10)
d <- rep(c(1, 0), each = 20)
y <- cbind(
  y1 = c(rbeta(20, 2, 6), rbeta(20, 4, 4)),
  y2 = c(rbeta(20, 5, 3), rbeta(20, 3, 5))
)
```

```
)
cmr_multiple_outcomes(y, d, weights = c(0.6, 0.4))
```

cmr_proxy

Proxy or delayed-outcome CMR assignment

Description

Estimate a proxy-outcome rectangle, widen it using the bridge radius zeta, and return the CMR treatment share for the primary-outcome design.

Usage

```
cmr_proxy(
  proxy_y,
  d,
  zeta,
  alpha = 0.05,
  method = c("auto", "bounded", "bernoulli", "maurer_pontil", "mp", "bernoulli_exact",
    "martinez_taboada_ramdas", "mtr"),
  beta = NULL,
  correction = c("bonferroni", "sidak_arms"),
  normalize = FALSE,
  lower = NULL,
  upper = NULL,
  na.rm = TRUE,
  tol = 1e-11
)

cmr_delayed_outcome(
  proxy_y,
  d,
  zeta,
  alpha = 0.05,
  method = c("auto", "bounded", "bernoulli", "maurer_pontil", "mp", "bernoulli_exact",
    "martinez_taboada_ramdas", "mtr"),
  beta = NULL,
  correction = c("bonferroni", "sidak_arms"),
  normalize = FALSE,
  lower = NULL,
  upper = NULL,
  na.rm = TRUE,
  tol = 1e-11
)
```

Arguments

proxy_y	Pilot proxy or delayed-primary outcomes.
d	Pilot treatment indicator; treatment is 1 and control is 0.
zeta	Nonnegative standard-deviation bridge radius. Provide a scalar shared across arms or a treatment/control pair.
alpha	Target joint error level.
method	Confidence-set method. "auto" chooses exact Bernoulli bounds for 0/1 outcomes and bounded Maurer-Pontil bounds otherwise.
beta	Optional endpoint error allocation. If NULL, error is split according to correction.
correction	Endpoint error correction, either "bonferroni" or "sidak_arms".
normalize	If TRUE, normalize bounded proxy outcomes to $[0, 1]$ before computing variances.
lower, upper	Optional lower and upper outcome bounds used when normalize = TRUE.
na.rm	If TRUE, drop rows with missing proxy_y or d.
tol	Numerical tolerance for exact Bernoulli bound inversion.

Value

A list of class `cmr_proxy` and `cmr_two_arm` with treatment share `pi`, CMR certificate `U_CMR`, widened confidence rectangle, pilot summaries, `zeta`, bridge diagnostics, endpoint error allocation, and method metadata.

See Also

Other CMR rules: [binary_rectangle_corners\(\)](#), [cmr_multiarm\(\)](#), [cmr_multiarm_from_rectangle\(\)](#), [cmr_multiple_outcomes\(\)](#), [cmr_stratified\(\)](#), [cmr_stratified_from_rectangle\(\)](#), [cmr_two_arm\(\)](#), [cmr_unbounded\(\)](#), [cmr_unbounded_from_rectangle\(\)](#), [print.cmr_two_arm\(\)](#)

Examples

```
set.seed(12)
d <- rep(c(1, 0), each = 30)
proxy_y <- c(rbeta(30, 2, 6), rbeta(30, 4, 4))
cmr_proxy(proxy_y, d, zeta = 0.05)
```

cmr_stratified

Stratified CMR assignment

Description

Estimate cell-specific variance confidence intervals from pilot data and return the stratified CMR assignment across treatment/control by stratum cells.

Usage

```
cmr_stratified(
  y,
  d,
  strata,
  strata_share,
  alpha = 0.05,
  method = c("auto", "bounded", "bernoulli", "maurer_pontil", "mp", "bernoulli_exact",
    "martinez_taboada_ramdas", "mtr"),
  beta = NULL,
  normalize = FALSE,
  lower = NULL,
  upper = NULL,
  na.rm = TRUE,
  tol = 1e-11,
  solver_control = list(),
  max_vertices = 65536L
)
```

Arguments

y	Pilot outcomes.
d	Pilot treatment indicator; treatment is 1 and control is 0.
strata	Pilot stratum labels.
strata_share	Named stratum population shares that sum to one.
alpha	Target joint error level.
method	Confidence-set method. "auto" chooses exact Bernoulli bounds for 0/1 outcomes and bounded Maurer-Pontil bounds otherwise.
beta	Optional endpoint error allocation. If NULL, Bonferroni error is split across all lower and upper treatment/control by stratum endpoints.
normalize	If TRUE, normalize bounded outcomes to $[0, 1]$ before computing variances.
lower, upper	Optional lower and upper outcome bounds used when <code>normalize = TRUE</code> .
na.rm	If TRUE, drop rows with missing y, d, or strata.
tol	Numerical tolerance for exact Bernoulli bound inversion.
solver_control	Optional list of solver controls for the general vertex epigraph solver.
max_vertices	Maximum number of hyperrectangle vertices to enumerate.

Value

A list of class `cmr_stratified` with total cell assignment shares `pi`, matrix form `pi_matrix`, sampling and treatment margins, CMR certificate `U_CMR`, confidence set, pilot summaries, endpoint error allocation, and diagnostics.

See Also

Other CMR rules: `binary_rectangle_corners()`, `cmr_multiarm()`, `cmr_multiarm_from_rectangle()`, `cmr_multiple_outcomes()`, `cmr_proxy()`, `cmr_stratified_from_rectangle()`, `cmr_two_arm()`, `cmr_unbounded()`, `cmr_unbounded_from_rectangle()`, `print.cmr_two_arm()`

Examples

```
set.seed(7)
strata <- rep(c("A", "B"), each = 40)
d <- rep(rep(c(1, 0), each = 20), 2)
y <- c(rbeta(20, 2, 6), rbeta(20, 4, 4),
      rbeta(20, 5, 3), rbeta(20, 3, 5))
cmr_stratified(y, d, strata, strata_share = c(A = 0.45, B = 0.55))
```

cmr_stratified_from_rectangle

Stratified CMR from a variance rectangle

Description

Compute a stratified CMR allocation from a supplied cell-level variance rectangle.

Usage

```
cmr_stratified_from_rectangle(
  rectangle,
  strata_share,
  control = list(),
  max_vertices = 65536L
)
```

Arguments

<code>rectangle</code>	Stratified variance rectangle, a list with lower and upper $2 \times S$ matrices.
<code>strata_share</code>	Named stratum population shares that sum to one.
<code>control</code>	Optional list of solver controls for the general vertex epigraph solver.
<code>max_vertices</code>	Maximum number of hyperrectangle vertices to enumerate.

Value

A list of class `cmr_stratified` with assignment shares `pi`, `pi_matrix`, stratum sampling margins, within-stratum treatment margins, CMR certificate `U_CMR`, checked rectangle, vertex diagnostics, and solver diagnostics.

See Also

Other CMR rules: `binary_rectangle_corners()`, `cmr_multiarm()`, `cmr_multiarm_from_rectangle()`, `cmr_multiple_outcomes()`, `cmr_proxy()`, `cmr_stratified()`, `cmr_two_arm()`, `cmr_unbounded()`, `cmr_unbounded_from_rectangle()`, `print.cmr_two_arm()`

Other rectangle helpers: `binary_rectangle_corners()`, `cmr_multiarm_from_rectangle()`, `cmr_unbounded_from_rect`, `folded_binomial_pmf()`, `multiarm_variance_objective()`, `rectangle_bernoulli_binary()`, `rectangle_bounded_two_arm()`, `rectangle_multiarm()`, `rectangle_multiple_outcomes()`, `rectangle_proxy()`, `rectangle_stratified()`, `rectangle_unbounded()`, `stratified_variance_objective()`, `variance_bounds_bernoulli_exact()`, `variance_bounds_maurer_pontil()`, `variance_bounds_unbounded_mom()`

Examples

```
strata_share <- c(A = 0.4, B = 0.6)
rect <- list(
  lower = rbind(treatment = c(A = 0.01, B = 0.04),
                control = c(A = 0.02, B = 0.03)),
  upper = rbind(treatment = c(A = 0.08, B = 0.12),
                control = c(A = 0.09, B = 0.10))
)
cmr_stratified_from_rectangle(rect, strata_share)
```

cmr_two_arm

Two-arm Conditional Minimax Regret assignment

Description

Estimate a finite-sample variance confidence rectangle from pilot data and return the two-arm Conditional Minimax Regret (CMR) assignment.

Usage

```
cmr_two_arm(
  y,
  d,
  alpha = 0.05,
  method = c("auto", "bounded", "bernoulli", "maurer_pontil", "mp", "bernoulli_exact",
             "martinez_taboada_ramdas", "mtr", "unbounded", "unbounded_mom", "median_of_means",
             "mom"),
  beta = NULL,
  correction = c("bonferroni", "sidak_arms"),
  normalize = FALSE,
  lower = NULL,
  upper = NULL,
  psi = NULL,
  na.rm = TRUE,
  tol = 1e-11
```

```

)

cmr_binary(
  y,
  d,
  alpha = 0.05,
  method = c("auto", "bounded", "bernoulli", "maurer_pontil", "mp", "bernoulli_exact",
    "martinez_taboada_ramdas", "mtr", "unbounded", "unbounded_mom", "median_of_means",
    "mom"),
  beta = NULL,
  correction = c("bonferroni", "sidak_arms"),
  normalize = FALSE,
  lower = NULL,
  upper = NULL,
  psi = NULL,
  na.rm = TRUE,
  tol = 1e-11
)

```

Arguments

y	Pilot outcomes. For bounded and Bernoulli methods, outcomes must be in $[0, 1]$ unless <code>normalize = TRUE</code> . For unbounded methods, outcomes are raw numeric values and <code>psi</code> is required.
d	Pilot treatment indicator; treatment is 1 and control is 0.
alpha	Target joint error level for the variance confidence set.
method	Confidence-set method. "auto" uses exact Bernoulli bounds for 0/1 outcomes and bounded Maurer-Pontil bounds otherwise. "bounded", "maurer_pontil", and "mp" are synonyms. "bernoulli" and "bernoulli_exact" use folded-binomial exact bounds. "mtr" and "martinez_taboada_ramdas" use the empirical-Bernstein MTR bounds. "unbounded", "unbounded_mom", "median_of_means", and "mom" dispatch to the unbounded-outcome median-of-means extension.
beta	Optional endpoint error allocation. If NULL, error is split across lower and upper endpoints using correction.
correction	Endpoint error correction, either "bonferroni" or "sidak_arms" for two-arm bounded/Bernoulli/proxy workflows.
normalize	If TRUE, normalize bounded outcomes to $[0, 1]$ before computing the rectangle.
lower, upper	Optional lower and upper outcome bounds used when <code>normalize = TRUE</code> .
psi	Bounded-kurtosis parameter for unbounded-outcome methods. Provide a scalar or a treatment/control pair.
na.rm	If TRUE, drop rows with missing y or d.
tol	Numerical tolerance for exact Bernoulli bound inversion.

Value

A list of class `cmr_two_arm` with treatment share `pi`, regret certificate `U_CMR`, confidence rectangle, pilot summaries, endpoint error allocation, and diagnostics. The object has compact `print()` and `summary()` methods.

See Also

Other CMR rules: `binary_rectangle_corners()`, `cmr_multiarm()`, `cmr_multiarm_from_rectangle()`, `cmr_multiple_outcomes()`, `cmr_proxy()`, `cmr_stratified()`, `cmr_stratified_from_rectangle()`, `cmr_unbounded()`, `cmr_unbounded_from_rectangle()`, `print.cmr_two_arm()`

Examples

```
set.seed(1)
d <- rep(c(1, 0), each = 40)
y <- c(rbeta(40, 2, 6), rbeta(40, 4, 4))

fit <- cmr_two_arm(y, d, alpha = 0.05, method = "bounded")
fit
summary(fit)
```

cmr_unbounded

Unbounded-outcome CMR assignment

Description

Estimate a median-of-means variance rectangle from raw pilot outcomes and return the unbounded-outcome CMR assignment.

Usage

```
cmr_unbounded(y, d, psi = NULL, alpha = 0.05, na.rm = TRUE)
```

Arguments

<code>y</code>	Pilot outcomes.
<code>d</code>	Pilot treatment indicator; treatment is 1 and control is 0.
<code>psi</code>	Bounded-kurtosis parameter, either a scalar shared across arms or a treatment/control pair.
<code>alpha</code>	Target joint error level.
<code>na.rm</code>	If TRUE, drop rows with missing <code>y</code> or <code>d</code> .

Value

A list of class `cmr_unbounded` and `cmr_two_arm`. If both one-arm bounds are active, the object contains `pi`, finite `U_CMR`, rectangle, pilot summaries, and diagnostics. If the pilot is inactive, the function returns balance (`pi = 0.5`) with `U_CMR = Inf` and diagnostics explaining the fallback.

See Also

Other CMR rules: `binary_rectangle_corners()`, `cmr_multiarm()`, `cmr_multiarm_from_rectangle()`, `cmr_multiple_outcomes()`, `cmr_proxy()`, `cmr_stratified()`, `cmr_stratified_from_rectangle()`, `cmr_two_arm()`, `cmr_unbounded_from_rectangle()`, `print.cmr_two_arm()`

Examples

```
set.seed(4)
d <- rep(c(1, 0), each = 220)
y <- c(rnorm(220, sd = 1.3), rnorm(220, sd = 0.8))
cmr_unbounded(y, d, psi = 3)
```

`cmr_unbounded_from_rectangle`

Unbounded-outcome CMR from a variance rectangle

Description

Compute the closed-form two-arm CMR allocation for a supplied nonnegative variance rectangle for unbounded outcomes.

Usage

```
cmr_unbounded_from_rectangle(rectangle)
```

Arguments

`rectangle` Two-arm nonnegative variance rectangle with names `v_l1`, `v_u1`, `v_l0`, and `v_u0`.

Value

A list of class `cmr_unbounded` and `cmr_two_arm` with treatment share `pi`, CMR certificate `U_CMR`, rectangle, corner regrets, binding diagnostics, and method metadata.

See Also

Other CMR rules: `binary_rectangle_corners()`, `cmr_multiarm()`, `cmr_multiarm_from_rectangle()`, `cmr_multiple_outcomes()`, `cmr_proxy()`, `cmr_stratified()`, `cmr_stratified_from_rectangle()`, `cmr_two_arm()`, `cmr_unbounded()`, `print.cmr_two_arm()`

Other rectangle helpers: `binary_rectangle_corners()`, `cmr_multiarm_from_rectangle()`, `cmr_stratified_from_rectangle()`, `folded_binomial_pmf()`, `multiarm_variance_objective()`, `rectangle_bernoulli_binary()`, `rectangle_bounded_two_arm()`, `rectangle_multiarm()`, `rectangle_multiple_outcomes()`, `rectangle_proxy()`, `rectangle_stratified()`, `rectangle_unbounded()`, `stratified_variance_objective()`, `variance_bounds_bernoulli_exact()`, `variance_bounds_maurer_pontil()`, `variance_bounds_unbounded_mom()`

Examples

```
rect <- c(v_l1 = 0.5, v_u1 = 1.4, v_l0 = 0.2, v_u0 = 1.0)
cmr_unbounded_from_rectangle(rect)
```

coverage_indicator	<i>Diagnostic metrics for CMR simulations</i>
--------------------	---

Description

Small helpers for checking rectangle coverage, certificate validity, boundary assignment, and realized variance gains in simulation or validation code.

Usage

```
coverage_indicator(rectangle, truth)

certificate_valid(pi, U, truth, tol = 1e-10)

boundary_indicator(pi, tol = 0)

boundary_rate(pi, tol = 0)

saving_vs_balance(pi, v1, v0)

share_of_oracle_gain(pi, v1, v0, tol = 1e-12)
```

Arguments

rectangle	Two-arm variance rectangle, either a numeric vector with names <code>v_l1</code> , <code>v_u1</code> , <code>v_l0</code> , <code>v_u0</code> or a rectangle object returned by a two-arm rectangle constructor.
truth	Named true variances, with entries <code>v1</code> and <code>v0</code> .
pi	Treatment assignment share.
U	CMR certificate to check against realized regret.
tol	Nonnegative numerical tolerance.
v1	Treatment-arm variance.
v0	Control-arm variance.

Value

`coverage_indicator()`, `certificate_valid()`, and `boundary_indicator()` return logical values. `boundary_rate()`, `saving_vs_balance()`, and `share_of_oracle_gain()` return numeric values.

Examples

```
rect <- c(v_l1 = 0.02, v_u1 = 0.12, v_l0 = 0.01, v_u0 = 0.08)
truth <- c(v1 = 0.09, v0 = 0.04)
fit <- cmr_two_arm_from_rectangle(rect)
coverage_indicator(rect, truth)
certificate_valid(fit$pi, fit$U_CMR, truth)
saving_vs_balance(fit$pi, truth["v1"], truth["v0"])
```

folded_binomial_pmf	<i>Folded-binomial distribution for Bernoulli sample variances</i>
---------------------	--

Description

Probability mass and tail probabilities for the folded count that determines the exact Bernoulli sample variance.

Usage

```
folded_binomial_pmf(v, m)

folded_binomial_tails(v, m)
```

Arguments

v	Bernoulli variance in $[0, 1/4]$.
m	Arm sample size, at least 2.

Value

folded_binomial_pmf() returns a named probability vector over folded counts $0, \dots, \text{floor}(m / 2)$.
 folded_binomial_tails() returns a list with lower and upper cumulative tail probabilities.

See Also

Other rectangle helpers: [binary_rectangle_corners\(\)](#), [cmr_multiarm_from_rectangle\(\)](#), [cmr_stratified_from_rectangle\(\)](#), [cmr_unbounded_from_rectangle\(\)](#), [multiarm_variance_objective\(\)](#), [rectangle_bernoulli_binary\(\)](#), [rectangle_bounded_two_arm\(\)](#), [rectangle_multiarm\(\)](#), [rectangle_multiple_outcomes\(\)](#), [rectangle_proxy\(\)](#), [rectangle_stratified\(\)](#), [rectangle_unbounded\(\)](#), [stratified_variance_objective\(\)](#), [variance_bounds_bernoulli_exact\(\)](#), [variance_bounds_maurer_pontil\(\)](#), [variance_bounds_unbounded_mom\(\)](#)

Examples

```
folded_binomial_pmf(v = 0.20, m = 8)
folded_binomial_tails(v = 0.20, m = 8)
```

multiarm_variance_objective

Multi-arm variance objectives and Neyman allocation

Description

Helper functions for shared-control multi-arm variance objectives, oracle values, Neyman allocations, regret, and rectangle vertices.

Usage

```
multiarm_variance_objective(pi, variances)

multiarm_oracle_variance(variances)

assign_multiarm_neyman(variances)

multiarm_regret(pi, variances)

multiarm_rectangle_vertices(rectangle, max_vertices = 65536L)
```

Arguments

pi	Named assignment-share vector over all arms, including control arm "0".
variances	Named variance vector over all arms, including control arm "0".
rectangle	Multi-arm variance rectangle, either a matrix/data frame with lower and upper columns or a named vector with entries like v_l0, v_u0, v_l1, v_u1.
max_vertices	Maximum number of hyperrectangle vertices to enumerate.

Value

Numeric objective/regret values, named assignment vectors, or a vertex matrix. `assign_multiarm_neyman()` returns total assignment shares over control and all treatment arms. `multiarm_rectangle_vertices()` returns one row per variance-rectangle vertex.

See Also

Other assignment helpers: [assign_balance\(\)](#), [stratified_variance_objective\(\)](#), [variance_objective\(\)](#)

Other rectangle helpers: [binary_rectangle_corners\(\)](#), [cmr_multiarm_from_rectangle\(\)](#), [cmr_stratified_from_rect](#), [cmr_unbounded_from_rectangle\(\)](#), [folded_binomial_pmf\(\)](#), [rectangle_bernoulli_binary\(\)](#), [rectangle_bounded_two_arm\(\)](#), [rectangle_multiarm\(\)](#), [rectangle_multiple_outcomes\(\)](#), [rectangle_proxy\(\)](#), [rectangle_stratified\(\)](#), [rectangle_unbounded\(\)](#), [stratified_variance_objective\(\)](#), [variance_bounds_bernoulli_exact\(\)](#), [variance_bounds_maurer_pontil\(\)](#), [variance_bounds_unbounded_mom\(\)](#)

Examples

```
variances <- c("0" = 0.05, "1" = 0.10, "2" = 0.04)
pi <- assign_multiarm_neyman(variances)
multiarm_variance_objective(pi, variances)
multiarm_regret(pi, variances)
```

pilot_plan

*Pilot/main-wave planning summary***Description**

Summarize the pilot-size viability band and a default pilot-size suggestion.

Usage

```
pilot_plan(
  n,
  sigma1,
  sigma0,
  alpha = 0.05,
  method = c("bounded", "bernoulli"),
  input = c("sd", "variance"),
  accounting = c("design_only", "pooled"),
  desired_pilot = NULL,
  strict_upper = TRUE
)
```

```
cmr_plan(
  n,
  sigma1,
  sigma0,
  alpha = 0.05,
  method = c("bounded", "bernoulli"),
  input = c("sd", "variance"),
  accounting = c("design_only", "pooled"),
  desired_pilot = NULL,
  strict_upper = TRUE
)
```

Arguments

n	Total experimental sample size across pilot and main wave.
sigma1	Treatment-arm standard deviation, or variance when input = "variance".
sigma0	Control-arm standard deviation, or variance when input = "variance".
alpha	Target error level.

method	Planning method, either "bounded" or "bernoulli".
input	Whether sigma1 and sigma0 are standard deviations ("sd") or variances ("variance").
accounting	"design_only" applies the break-even pilot-share cap; "pooled" ignores that cap.
desired_pilot	Optional user-proposed pilot size to classify.
strict_upper	If TRUE, feasible pilot sizes must be strictly below the design-only break-even cap.

Value

A list of class `cmr_pilot_plan` with `band`, `suggested_pilot`, `default_two_thirds_power`, `desired_pilot`, `desired_status`, a text recommendation, and a caveat that the screen is necessary rather than sufficient.

See Also

Other pilot planning: [activation_threshold_bounded\(\)](#), [break_even_pilot_share\(\)](#), [pilot_viability_band\(\)](#)

Examples

```
cmr_plan(
  n = 1000,
  sigma1 = 0.35,
  sigma0 = 0.20,
  desired_pilot = 100
)
```

pilot_viability_band *Pilot viability band*

Description

Compute a necessary viability screen for pilot sizes before choosing a pilot/main-wave split.

Usage

```
pilot_viability_band(
  n,
  sigma1,
  sigma0,
  alpha = 0.05,
  method = c("bounded", "bernoulli"),
  input = c("sd", "variance"),
  accounting = c("design_only", "pooled"),
  strict_upper = TRUE
)
```

Arguments

n	Total experimental sample size across pilot and main wave.
sigma1	Treatment-arm standard deviation, or variance when input = "variance".
sigma0	Control-arm standard deviation, or variance when input = "variance".
alpha	Target error level.
method	Planning method, either "bounded" or "bernoulli".
input	Whether sigma1 and sigma0 are standard deviations ("sd") or variances ("variance").
accounting	"design_only" applies the break-even pilot-share cap; "pooled" ignores that cap.
strict_upper	If TRUE, feasible pilot sizes must be strictly below the design-only break-even cap.

Value

A list of class `cmr_pilot_viability_band` with total sample size, standard deviations, method, break-even share and total, activation threshold, feasible even pilot sizes, and min/max feasible pilot sizes.

See Also

Other pilot planning: [activation_threshold_bounded\(\)](#), [break_even_pilot_share\(\)](#), [pilot_plan\(\)](#)

Examples

```
pilot_viability_band(n = 1000, sigma1 = 0.35, sigma0 = 0.20)
```

```
print.cmr_two_arm      Print and summarize CMR results
```

Description

Compact display methods for CMR result objects. These methods show the allocation, CMR certificate, method, sample size, and status without dumping nested confidence-set internals.

Usage

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

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

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



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

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

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

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

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

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

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

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

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

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

Arguments

x, object	A CMR result object or summary object.
...	Reserved for future extensions.

Value

print() methods return the original object invisibly. summary() methods return a compact list of class summary.cmr_result.

See Also

Other CMR rules: [binary_rectangle_corners\(\)](#), [cmr_multiarm\(\)](#), [cmr_multiarm_from_rectangle\(\)](#), [cmr_multiple_outcomes\(\)](#), [cmr_proxy\(\)](#), [cmr_stratified\(\)](#), [cmr_stratified_from_rectangle\(\)](#), [cmr_two_arm\(\)](#), [cmr_unbounded\(\)](#), [cmr_unbounded_from_rectangle\(\)](#)

Examples

```

set.seed(13)
d <- rep(c(1, 0), each = 20)
y <- c(rbeta(20, 2, 6), rbeta(20, 4, 4))
fit <- cmr_two_arm(y, d)
print(fit)
summary(fit)

```

rectangle_bernoulli_binary

Two-arm confidence rectangles

Description

Construct a two-arm variance confidence rectangle from pilot data. These are expert helpers used by `cmr_two_arm()` and related extension functions.

Usage

```

rectangle_bernoulli_binary(
  y,
  d,
  alpha = 0.05,
  beta = NULL,
  correction = c("bonferroni", "sidak_arms"),
  na.rm = TRUE,
  tol = 1e-11
)

rectangle_binary(
  y,
  d,
  alpha = 0.05,
  method = c("auto", "bounded", "bernoulli", "maurer_pontil", "mp", "bernoulli_exact",
    "martinez_taboada_ramdas", "mtr", "unbounded", "unbounded_mom", "median_of_means",
    "mom"),
  beta = NULL,
  correction = c("bonferroni", "sidak_arms"),
  normalize = FALSE,
  lower = NULL,
  upper = NULL,
  psi = NULL,
  na.rm = TRUE,
  tol = 1e-11
)

```

```

rectangle_two_arm(
  y,
  d,
  alpha = 0.05,
  method = c("auto", "bounded", "bernoulli", "maurer_pontil", "mp", "bernoulli_exact",
    "martinez_taboada_ramdas", "mtr", "unbounded", "unbounded_mom", "median_of_means",
    "mom"),
  beta = NULL,
  correction = c("bonferroni", "sidak_arms"),
  normalize = FALSE,
  lower = NULL,
  upper = NULL,
  psi = NULL,
  na.rm = TRUE,
  tol = 1e-11
)

rectangle_bernoulli_two_arm(
  y,
  d,
  alpha = 0.05,
  beta = NULL,
  correction = c("bonferroni", "sidak_arms"),
  na.rm = TRUE,
  tol = 1e-11
)

```

Arguments

y	Pilot outcomes.
d	Pilot treatment indicator; treatment is 1 and control is 0.
alpha	Target joint error level.
beta	Optional endpoint error allocation. If NULL, error is split according to correction.
correction	Endpoint error correction, either "bonferroni" or "sidak_arms" for two-arm workflows.
na.rm	If TRUE, drop rows with missing y or d.
tol	Numerical tolerance for exact Bernoulli bound inversion.
method	Confidence-set method. "auto" chooses exact Bernoulli bounds for 0/1 outcomes and bounded Maurer-Pontil bounds otherwise.
normalize	If TRUE, normalize bounded outcomes to $[0, 1]$ before computing variances.
lower, upper	Optional lower and upper outcome bounds used when normalize = TRUE.
psi	Bounded-kurtosis parameter used only when method is an unbounded-outcome method.

Value

A rectangle object. Bounded and Bernoulli methods return a `cmr_binary_rectangle`; unbounded methods return a `cmr_unbounded_rectangle`. The object contains the numeric rectangle, one-arm bound details, endpoint error allocation, pilot sample sizes and variance estimates, and method metadata.

See Also

Other rectangle helpers: `binary_rectangle_corners()`, `cmr_multiarm_from_rectangle()`, `cmr_stratified_from_rect`, `cmr_unbounded_from_rectangle()`, `folded_binomial_pmf()`, `multiarm_variance_objective()`, `rectangle_bounded_two_arm()`, `rectangle_multiarm()`, `rectangle_multiple_outcomes()`, `rectangle_proxy()`, `rectangle_stratified()`, `rectangle_unbounded()`, `stratified_variance_objective()`, `variance_bounds_bernoulli_exact()`, `variance_bounds_maurer_pontil()`, `variance_bounds_unbounded_mom()`

Examples

```
d <- rep(c(1, 0), each = 6)
y <- c(1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1)
rectangle_two_arm(y, d, method = "bernoulli")
rectangle_binary(y, d, method = "auto")
```

`rectangle_bounded_two_arm`

Bounded two-arm confidence rectangle

Description

Construct a two-arm variance confidence rectangle for bounded outcomes using Maurer-Pontil or Martinez-Taboada-Ramdas one-arm bounds.

Usage

```
rectangle_bounded_two_arm(
  y,
  d,
  alpha = 0.05,
  method = c("bounded", "maurer_pontil", "mp", "martinez_taboada_ramdas", "mtr"),
  beta = NULL,
  correction = c("bonferroni", "sidak_arms"),
  normalize = FALSE,
  lower = NULL,
  upper = NULL,
  na.rm = TRUE
)

rectangle_bounded_binary(
```

```

y,
d,
alpha = 0.05,
method = c("bounded", "maurer_pontil", "mp", "martinez_taboada_ramdas", "mtr"),
beta = NULL,
correction = c("bonferroni", "sidak_arms"),
normalize = FALSE,
lower = NULL,
upper = NULL,
na.rm = TRUE
)

```

Arguments

y	Pilot outcomes.
d	Pilot treatment indicator; treatment is 1 and control is 0.
alpha	Target joint error level.
method	Bounded-outcome method. "bounded", "maurer_pontil", and "mp" are synonyms; "martinez_taboada_ramdas" and "mtr" use MTR bounds.
beta	Optional endpoint error allocation. If NULL, error is split according to correction.
correction	Endpoint error correction, either "bonferroni" or "sidak_arms".
normalize	If TRUE, normalize outcomes to $[0, 1]$ before computing variances.
lower, upper	Optional lower and upper outcome bounds used when <code>normalize = TRUE</code> .
na.rm	If TRUE, drop rows with missing y or d.

Value

A `cmr_binary_rectangle` list with rectangle, one-arm bound objects for treatment and control, endpoint error allocation, sample sizes, pilot variance estimates, normalization details, and method metadata.

See Also

Other rectangle helpers: [binary_rectangle_corners\(\)](#), [cmr_multiarm_from_rectangle\(\)](#), [cmr_stratified_from_rectangle\(\)](#), [cmr_unbounded_from_rectangle\(\)](#), [folded_binomial_pmf\(\)](#), [multiarm_variance_objective\(\)](#), [rectangle_bernoulli_binary\(\)](#), [rectangle_multiarm\(\)](#), [rectangle_multiple_outcomes\(\)](#), [rectangle_proxy\(\)](#), [rectangle_stratified\(\)](#), [rectangle_unbounded\(\)](#), [stratified_variance_objective\(\)](#), [variance_bounds_bernoulli_exact\(\)](#), [variance_bounds_maurer_pontil\(\)](#), [variance_bounds_unbounded_mom\(\)](#)

Examples

```

d <- rep(c(1, 0), each = 5)
y <- c(0.20, 0.40, 0.30, 0.10, 0.60, 0.50, 0.30, 0.20, 0.40, 0.10)
rectangle_bounded_binary(y, d, method = "bounded")

```

rectangle_multiarm *Multi-arm confidence rectangle*

Description

Construct arm-specific variance confidence intervals for a shared-control multi-arm design.

Usage

```
rectangle_multiarm(
  y,
  arm,
  alpha = 0.05,
  method = c("auto", "bounded", "bernoulli", "maurer_pontil", "mp", "bernoulli_exact",
    "martinez_taboada_ramdas", "mtr"),
  beta = NULL,
  control_arm = 0,
  normalize = FALSE,
  lower = NULL,
  upper = NULL,
  na.rm = TRUE,
  tol = 1e-11
)
```

Arguments

y	Pilot outcomes.
arm	Pilot arm labels. The control arm is identified by control_arm and internally standardized to "0".
alpha	Target joint error level.
method	Confidence-set method. "auto" chooses exact Bernoulli bounds for 0/1 outcomes and bounded Maurer-Pontil bounds otherwise.
beta	Optional endpoint error allocation. If NULL, Bonferroni error is split across all lower and upper arm endpoints. A scalar, matrix, or named vector allocation may also be supplied.
control_arm	Label identifying the control arm in arm.
normalize	If TRUE, normalize bounded outcomes to [0, 1] before computing variances.
lower, upper	Optional lower and upper outcome bounds used when normalize = TRUE.
na.rm	If TRUE, drop rows with missing y or arm.
tol	Numerical tolerance for exact Bernoulli bound inversion.

Value

A list of class `cmr_multiarm_rectangle` with checked rectangle, arm labels, one-arm bound results, endpoint error allocation, sample sizes, pilot variance estimates, normalization details, and method metadata.

See Also

Other rectangle helpers: `binary_rectangle_corners()`, `cmr_multiarm_from_rectangle()`, `cmr_stratified_from_rec`, `cmr_unbounded_from_rectangle()`, `folded_binomial_pmf()`, `multiarm_variance_objective()`, `rectangle_bernoulli_binary()`, `rectangle_bounded_two_arm()`, `rectangle_multiple_outcomes()`, `rectangle_proxy()`, `rectangle_stratified()`, `rectangle_unbounded()`, `stratified_variance_objective()`, `variance_bounds_bernoulli_exact()`, `variance_bounds_maurer_pontil()`, `variance_bounds_unbounded_mom()`

Examples

```
set.seed(6)
arm <- rep(c(0, 1, 2), each = 12)
y <- c(rbeta(12, 4, 4), rbeta(12, 2, 6), rbeta(12, 5, 3))
rectangle_multiarm(y, arm, method = "bounded")
```

rectangle_multiple_outcomes

Multiple-outcome confidence rectangle

Description

Construct an effective two-arm variance rectangle for multiple outcomes. For `estimand = "index"`, outcomes are first combined using weights. For `estimand = "coprimary"`, one rectangle is built per outcome and combined conservatively using the outcome weights.

Usage

```
rectangle_multiple_outcomes(
  y,
  d,
  weights = NULL,
  estimand = c("coprimary", "index"),
  alpha = 0.05,
  method = c("auto", "bounded", "bernoulli", "maurer_pontil", "mp", "bernoulli_exact",
    "martinez_taboada_ramdas", "mtr"),
  beta = NULL,
  na.rm = TRUE,
  tol = 1e-11
)
```

Arguments

<code>y</code>	Pilot outcomes as a numeric matrix, data frame, or vector. Rows are units and columns are outcomes.
<code>d</code>	Pilot treatment indicator; treatment is 1 and control is 0.
<code>weights</code>	Optional nonnegative outcome weights. If <code>NULL</code> , equal weights are used.

estimand	Either "coprimary" or "index".
alpha	Target joint error level.
method	Confidence-set method. "auto" chooses exact Bernoulli bounds for 0/1 outcomes and bounded Maurer-Pontil bounds otherwise.
beta	Optional scalar endpoint error allocation.
na.rm	If TRUE, drop rows with missing y or d.
tol	Numerical tolerance for exact Bernoulli bound inversion.

Value

A list of class `cmr_multiple_outcomes_rectangle`. For `estimand = "index"`, this wraps the ordinary two-arm rectangle for the weighted index. For `estimand = "coprimary"`, it contains the effective two-arm rectangle, weights, outcome-specific bounds, pilot summaries, endpoint error allocation, and method metadata.

See Also

Other rectangle helpers: [binary_rectangle_corners\(\)](#), [cmr_multiarm_from_rectangle\(\)](#), [cmr_stratified_from_rectangle\(\)](#), [cmr_unbounded_from_rectangle\(\)](#), [folded_binomial_pmf\(\)](#), [multiarm_variance_objective\(\)](#), [rectangle_bernoulli_binary\(\)](#), [rectangle_bounded_two_arm\(\)](#), [rectangle_multiarm\(\)](#), [rectangle_proxy\(\)](#), [rectangle_stratified\(\)](#), [rectangle_unbounded\(\)](#), [stratified_variance_objective\(\)](#), [variance_bounds_bernoulli\(\)](#), [variance_bounds_maurer_pontil\(\)](#), [variance_bounds_unbounded_mom\(\)](#)

Examples

```
set.seed(9)
d <- rep(c(1, 0), each = 20)
y <- cbind(
  y1 = c(rbeta(20, 2, 6), rbeta(20, 4, 4)),
  y2 = c(rbeta(20, 5, 3), rbeta(20, 3, 5))
)
rectangle_multiple_outcomes(y, d, weights = c(0.6, 0.4))
```

rectangle_proxy	<i>Proxy or delayed-outcome confidence rectangle</i>
-----------------	--

Description

Construct a primary-outcome variance rectangle by estimating a proxy-outcome rectangle and widening each arm’s standard-deviation interval by ζ .

Usage

```

rectangle_proxy(
  proxy_y,
  d,
  zeta,
  alpha = 0.05,
  method = c("auto", "bounded", "bernoulli", "maurer_pontil", "mp", "bernoulli_exact",
    "martinez_taboada_ramdas", "mtr"),
  beta = NULL,
  correction = c("bonferroni", "sidak_arms"),
  normalize = FALSE,
  lower = NULL,
  upper = NULL,
  na.rm = TRUE,
  tol = 1e-11
)

rectangle_delayed_outcome(
  proxy_y,
  d,
  zeta,
  alpha = 0.05,
  method = c("auto", "bounded", "bernoulli", "maurer_pontil", "mp", "bernoulli_exact",
    "martinez_taboada_ramdas", "mtr"),
  beta = NULL,
  correction = c("bonferroni", "sidak_arms"),
  normalize = FALSE,
  lower = NULL,
  upper = NULL,
  na.rm = TRUE,
  tol = 1e-11
)

```

Arguments

proxy_y	Pilot proxy or delayed-primary outcomes.
d	Pilot treatment indicator; treatment is 1 and control is 0.
zeta	Nonnegative standard-deviation bridge radius. Provide a scalar shared across arms or a treatment/control pair.
alpha	Target joint error level.
method	Confidence-set method. "auto" chooses exact Bernoulli bounds for 0/1 outcomes and bounded Maurer-Pontil bounds otherwise.
beta	Optional endpoint error allocation. If NULL, error is split according to correction.
correction	Endpoint error correction, either "bonferroni" or "sidak_arms".
normalize	If TRUE, normalize bounded proxy outcomes to $[0, 1]$ before computing variances.

lower, upper	Optional lower and upper outcome bounds used when normalize = TRUE.
na.rm	If TRUE, drop rows with missing proxy_y or d.
tol	Numerical tolerance for exact Bernoulli bound inversion.

Value

A list of class `cmr_proxy_rectangle` and `cmr_binary_rectangle` with the widened primary-outcome rectangle, the underlying proxy confidence set, zeta, bridge metadata, endpoint error allocation, pilot summaries, and method metadata.

See Also

Other rectangle helpers: `binary_rectangle_corners()`, `cmr_multiarm_from_rectangle()`, `cmr_stratified_from_rectangle()`, `cmr_unbounded_from_rectangle()`, `folded_binomial_pmf()`, `multiarm_variance_objective()`, `rectangle_bernoulli_binary()`, `rectangle_bounded_two_arm()`, `rectangle_multiarm()`, `rectangle_multiple_outcome()`, `rectangle_stratified()`, `rectangle_unbounded()`, `stratified_variance_objective()`, `variance_bounds_bernoulli()`, `variance_bounds_maurer_pontil()`, `variance_bounds_unbounded_mom()`

Examples

```
set.seed(11)
d <- rep(c(1, 0), each = 30)
proxy_y <- c(rbeta(30, 2, 6), rbeta(30, 4, 4))
rectangle_proxy(proxy_y, d, zeta = 0.05)
```

rectangle_stratified *Stratified confidence rectangle*

Description

Construct cell-specific variance confidence intervals for a stratified two-arm design.

Usage

```
rectangle_stratified(
  y,
  d,
  strata,
  strata_share,
  alpha = 0.05,
  method = c("auto", "bounded", "bernoulli", "maurer_pontil", "mp", "bernoulli_exact",
    "martinez_taboada_ramdas", "mtr"),
  beta = NULL,
  normalize = FALSE,
  lower = NULL,
  upper = NULL,
```

```

    na.rm = TRUE,
    tol = 1e-11
  )

```

Arguments

y	Pilot outcomes.
d	Pilot treatment indicator; treatment is 1 and control is 0.
strata	Pilot stratum labels.
strata_share	Named stratum population shares that sum to one.
alpha	Target joint error level.
method	Confidence-set method. "auto" chooses exact Bernoulli bounds for 0/1 outcomes and bounded Maurer-Pontil bounds otherwise.
beta	Optional endpoint error allocation. If NULL, Bonferroni error is split across all lower and upper treatment/control by stratum endpoints.
normalize	If TRUE, normalize bounded outcomes to $[0, 1]$ before computing variances.
lower, upper	Optional lower and upper outcome bounds used when normalize = TRUE.
na.rm	If TRUE, drop rows with missing y, d, or strata.
tol	Numerical tolerance for exact Bernoulli bound inversion.

Value

A list of class `cmr_stratified_rectangle` with lower and upper rectangle matrices, checked rectangle details, stratum shares, cell-level one-arm bound results, endpoint error allocation, sample sizes, pilot variance estimates, normalization details, and method metadata.

See Also

Other rectangle helpers: [binary_rectangle_corners\(\)](#), [cmr_multiarm_from_rectangle\(\)](#), [cmr_stratified_from_rectangle\(\)](#), [cmr_unbounded_from_rectangle\(\)](#), [folded_binomial_pmf\(\)](#), [multiarm_variance_objective\(\)](#), [rectangle_bernoulli_binary\(\)](#), [rectangle_bounded_two_arm\(\)](#), [rectangle_multiarm\(\)](#), [rectangle_multiple_outcomes\(\)](#), [rectangle_proxy\(\)](#), [rectangle_unbounded\(\)](#), [stratified_variance_objective\(\)](#), [variance_bounds_bernoulli_exact\(\)](#), [variance_bounds_maurer_pontil\(\)](#), [variance_bounds_unbounded_mom\(\)](#)

Examples

```

set.seed(8)
strata <- rep(c("A", "B"), each = 24)
d <- rep(rep(c(1, 0), each = 12), 2)
y <- c(rbeta(12, 2, 6), rbeta(12, 4, 4),
      rbeta(12, 5, 3), rbeta(12, 3, 5))
rectangle_stratified(y, d, strata, strata_share = c(A = 0.45, B = 0.55))

```

rectangle_unbounded *Unbounded-outcome confidence rectangle*

Description

Construct a two-arm median-of-means variance rectangle for raw numeric outcomes under bounded-kurtosis inputs.

Usage

```
rectangle_unbounded(y, d, psi = NULL, alpha = 0.05, na.rm = TRUE)
```

Arguments

y	Pilot outcomes.
d	Pilot treatment indicator; treatment is 1 and control is 0.
psi	Bounded-kurtosis parameter, either a scalar shared across arms or a treatment/control pair.
alpha	Target joint error level.
na.rm	If TRUE, drop rows with missing y or d.

Value

A list of class `cmr_unbounded_rectangle` with `rectangle` when active, one-arm treatment and control bound objects, pilot summaries, block diagnostics, `psi`, and `status`. If either arm is inactive, `rectangle` is `NULL` and `status` explains why.

See Also

Other rectangle helpers: [binary_rectangle_corners\(\)](#), [cmr_multiarm_from_rectangle\(\)](#), [cmr_stratified_from_rectangle\(\)](#), [cmr_unbounded_from_rectangle\(\)](#), [folded_binomial_pmf\(\)](#), [multiarm_variance_objective\(\)](#), [rectangle_bernoulli_binary\(\)](#), [rectangle_bounded_two_arm\(\)](#), [rectangle_multiarm\(\)](#), [rectangle_multiple_outcomes\(\)](#), [rectangle_proxy\(\)](#), [rectangle_stratified\(\)](#), [stratified_variance_objective\(\)](#), [variance_bounds_bernoulli\(\)](#), [variance_bounds_maurer_pontil\(\)](#), [variance_bounds_unbounded_mom\(\)](#)

Examples

```
set.seed(3)
d <- rep(c(1, 0), each = 220)
y <- c(rnorm(220, sd = 1.3), rnorm(220, sd = 0.8))
rectangle_unbounded(y, d, psi = 3)
```

stratified_variance_objective

Stratified variance objectives and Neyman allocation

Description

Helper functions for stratified two-arm variance objectives, oracle values, Neyman allocations, regret, and rectangle vertices.

Usage

```
stratified_variance_objective(pi, variances, strata_share)

stratified_oracle_variance(variances, strata_share)

assign_stratified_neyman(variances, strata_share)

stratified_regret(pi, variances, strata_share)

stratified_rectangle_vertices(rectangle, strata_share, max_vertices = 65536L)
```

Arguments

pi	Assignment shares for each treatment/control by stratum cell. A vector should be named like "1:A" and "0:A", or a 2 x S matrix with treatment and control rows.
variances	Cell variances as a 2 x S matrix or data frame with treatment and control rows.
strata_share	Named stratum population shares that sum to one.
rectangle	Stratified variance rectangle, a list with lower and upper 2 x S matrices.
max_vertices	Maximum number of hyperrectangle vertices to enumerate.

Value

Numeric objective/regret values, named assignment vectors, or a vertex matrix. `assign_stratified_neyman()` returns total assignment shares over treatment/control by stratum cells.

See Also

Other assignment helpers: [assign_balance\(\)](#), [multiarm_variance_objective\(\)](#), [variance_objective\(\)](#)

Other rectangle helpers: [binary_rectangle_corners\(\)](#), [cmr_multiarm_from_rectangle\(\)](#), [cmr_stratified_from_rectangle\(\)](#), [cmr_unbounded_from_rectangle\(\)](#), [folded_binomial_pmf\(\)](#), [multiarm_variance_objective\(\)](#), [rectangle_bernoulli_binary\(\)](#), [rectangle_bounded_two_arm\(\)](#), [rectangle_multiarm\(\)](#), [rectangle_multiple_outcomes\(\)](#), [rectangle_proxy\(\)](#), [rectangle_stratified\(\)](#), [rectangle_unbounded\(\)](#), [variance_bounds_bernoulli_exact\(\)](#), [variance_bounds_maurer_pontil\(\)](#), [variance_bounds_unbounded_mom\(\)](#)

Examples

```
strata_share <- c(A = 0.4, B = 0.6)
variances <- rbind(
  treatment = c(A = 0.10, B = 0.04),
  control = c(A = 0.05, B = 0.08)
)
pi <- assign_stratified_neyman(variances, strata_share)
stratified_variance_objective(pi, variances, strata_share)
```

variance_bounds_bernoulli_exact

Exact Bernoulli variance bounds

Description

Exact one-arm variance confidence bounds for Bernoulli outcomes using the folded-binomial distribution of the sample variance.

Usage

```
variance_bounds_bernoulli_exact(y, beta_l, beta_u, na.rm = TRUE, tol = 1e-11)
```

Arguments

y	One-arm Bernoulli outcomes coded as 0 and 1.
beta_l	One-sided endpoint error for the lower variance bound.
beta_u	One-sided endpoint error for the upper variance bound.
na.rm	If TRUE, drop missing outcomes.
tol	Numerical tolerance for endpoint inversion.

Value

A list with lower bound L, upper bound U, folded-binomial variance estimate vhat, method name, sample size n, and folded-count details in statistic.

See Also

Other rectangle helpers: [binary_rectangle_corners\(\)](#), [cmr_multiarm_from_rectangle\(\)](#), [cmr_stratified_from_rectangle\(\)](#), [cmr_unbounded_from_rectangle\(\)](#), [folded_binomial_pmf\(\)](#), [multiarm_variance_objective\(\)](#), [rectangle_bernoulli_binary\(\)](#), [rectangle_bounded_two_arm\(\)](#), [rectangle_multiarm\(\)](#), [rectangle_multiple_outcomes\(\)](#), [rectangle_proxy\(\)](#), [rectangle_stratified\(\)](#), [rectangle_unbounded\(\)](#), [stratified_variance_objective\(\)](#), [variance_bounds_maurer_pontil\(\)](#), [variance_bounds_unbounded_mom\(\)](#)

Examples

```
y <- c(1, 0, 1, 1, 0, 0, 1, 0)
variance_bounds_bernoulli_exact(y, beta_l = 0.025, beta_u = 0.025)
```

variance_bounds_maurer_pontil

Bounded-outcome variance bounds

Description

One-arm distribution-free variance confidence bounds for outcomes in $[0, 1]$.

Usage

```
variance_bounds_maurer_pontil(y, beta_l, beta_u, na.rm = TRUE)
```

```
variance_bounds_martinez_taboada_ramdas(
  y,
  beta_l,
  beta_u,
  na.rm = TRUE,
  lower_alpha_split = 0.5,
  c1 = 0.5,
  c2 = 0.25^2,
  c3 = 0.25,
  c4 = 0.5,
  c5 = 2,
  cs = FALSE,
  tilde_cs = TRUE
)
```

Arguments

y	Pilot outcomes for one arm.
beta_l	One-sided endpoint error for the lower variance bound.
beta_u	One-sided endpoint error for the upper variance bound.
na.rm	If TRUE, drop missing outcomes.
lower_alpha_split	MTR split of the lower-tail error between variance and mean components.
c1, c2, c3, c4, c5	Martinez-Taboada-Ramdas tuning constants.
cs, tilde_cs	Logical flags for the MTR predictable-mixture variants.

Value

A list with lower bound L, upper bound U, sample variance vhat, method name, arm sample size n, and method-specific statistic details.

See Also

Other rectangle helpers: `binary_rectangle_corners()`, `cmr_multiarm_from_rectangle()`, `cmr_stratified_from_rect`, `cmr_unbounded_from_rectangle()`, `folded_binomial_pmf()`, `multiarm_variance_objective()`, `rectangle_bernoulli_binary()`, `rectangle_bounded_two_arm()`, `rectangle_multiarm()`, `rectangle_multiple_out`, `rectangle_proxy()`, `rectangle_stratified()`, `rectangle_unbounded()`, `stratified_variance_objective()`, `variance_bounds_bernoulli_exact()`, `variance_bounds_unbounded_mom()`

Examples

```
y <- c(0.10, 0.30, 0.40, 0.20, 0.70, 0.50)
variance_bounds_maurer_pontil(y, beta_l = 0.025, beta_u = 0.025)
```

variance_bounds_unbounded_mom

Unbounded-outcome median-of-means variance bounds

Description

Compute one-arm median-of-means variance bounds for raw numeric outcomes under a bounded-kurtosis input `psi`.

Usage

```
variance_bounds_unbounded_mom(y, alpha = 0.05, psi = NULL, na.rm = TRUE)
```

Arguments

<code>y</code>	One-arm pilot outcomes.
<code>alpha</code>	Target one-arm error level.
<code>psi</code>	Bounded-kurtosis parameter. Must be at least 1.
<code>na.rm</code>	If TRUE, drop missing outcomes.

Value

A list with lower bound `L`, upper bound `U`, median-of-means variance estimate `vhat`, method name, sample size `n`, activation flag `active`, status string, and block-level statistic details. If the pilot is too small or the relative-error radius is too large, `active = FALSE`, `L = NA`, and `U = Inf`.

See Also

Other rectangle helpers: `binary_rectangle_corners()`, `cmr_multiarm_from_rectangle()`, `cmr_stratified_from_rect`, `cmr_unbounded_from_rectangle()`, `folded_binomial_pmf()`, `multiarm_variance_objective()`, `rectangle_bernoulli_binary()`, `rectangle_bounded_two_arm()`, `rectangle_multiarm()`, `rectangle_multiple_out`, `rectangle_proxy()`, `rectangle_stratified()`, `rectangle_unbounded()`, `stratified_variance_objective()`, `variance_bounds_bernoulli_exact()`, `variance_bounds_maurer_pontil()`

Examples

```
set.seed(2)
y <- rnorm(200)
variance_bounds_unbounded_mom(y, alpha = 0.05, psi = 3)
```

variance_objective	<i>Two-arm variance objectives and Neyman allocation</i>
--------------------	--

Description

Helper functions for the two-arm variance objective, oracle variance, Neyman allocation, and regret. These are useful for checking CMR certificates and comparing CMR against oracle or feasible-Neyman benchmarks.

Usage

```
variance_objective(pi, v1, v0)
```

```
oracle_variance(v1, v0)
```

```
assign_neyman(v1, v0)
```

```
regret(pi, v1, v0)
```

Arguments

pi	Treatment assignment share. Values must lie in $[0, 1]$.
v1	Treatment-arm outcome variance. For bounded or Bernoulli outcomes, this must lie in $[0, 1/4]$.
v0	Control-arm outcome variance. For bounded or Bernoulli outcomes, this must lie in $[0, 1/4]$.

Value

Numeric vector after ordinary R recycling of pi, v1, and v0. `variance_objective()` returns $v1 / pi + v0 / (1 - pi)$, `oracle_variance()` returns the Neyman-oracle value $(\sqrt{v1} + \sqrt{v0})^2$, `assign_neyman()` returns the treatment share, and `regret()` returns excess variance relative to the oracle.

See Also

Other assignment helpers: [assign_balance\(\)](#), [multiarm_variance_objective\(\)](#), [stratified_variance_objective\(\)](#)

Examples

```
v1 <- 0.12
v0 <- 0.04
pi <- assign_neyman(v1, v0)
variance_objective(pi, v1, v0)
oracle_variance(v1, v0)
regret(0.5, v1, v0)
```

Index

- * **CMR rules**
 - binary_rectangle_corners, 5
 - cmr_multiarm, 7
 - cmr_multiarm_from_rectangle, 8
 - cmr_multiple_outcomes, 9
 - cmr_proxy, 11
 - cmr_stratified, 12
 - cmr_stratified_from_rectangle, 14
 - cmr_two_arm, 15
 - cmr_unbounded, 17
 - cmr_unbounded_from_rectangle, 18
 - print.cmr_two_arm, 24
- * **assignment helpers**
 - assign_balance, 4
 - multiarm_variance_objective, 21
 - stratified_variance_objective, 37
 - variance_objective, 41
- * **diagnostic helpers**
 - coverage_indicator, 19
- * **pilot planning**
 - activation_threshold_bounded, 3
 - break_even_pilot_share, 6
 - pilot_plan, 22
 - pilot_viability_band, 23
- * **rectangle helpers**
 - binary_rectangle_corners, 5
 - cmr_multiarm_from_rectangle, 8
 - cmr_stratified_from_rectangle, 14
 - cmr_unbounded_from_rectangle, 18
 - folded_binomial_pmf, 20
 - multiarm_variance_objective, 21
 - rectangle_bernoulli_binary, 26
 - rectangle_bounded_two_arm, 28
 - rectangle_multiarm, 30
 - rectangle_multiple_outcomes, 31
 - rectangle_proxy, 32
 - rectangle_stratified, 34
 - rectangle_unbounded, 36
 - stratified_variance_objective, 37
 - variance_bounds_bernoulli_exact, 38
 - variance_bounds_maurer_pontil, 39
 - variance_bounds_unbounded_mom, 40
- activation_threshold_bernoulli
(activation_threshold_bounded), 3
- activation_threshold_bounded, 3, 7, 23, 24
- assign_additive_regularized_neyman
(assign_balance), 4
- assign_balance, 4, 21, 37, 41
- assign_exponential_regularized_neyman
(assign_balance), 4
- assign_feasible_neyman
(assign_balance), 4
- assign_multiarm_balance
(assign_balance), 4
- assign_multiarm_neyman
(multiarm_variance_objective), 21
- assign_neyman (variance_objective), 41
- assign_stratified_balance
(assign_balance), 4
- assign_stratified_neyman
(stratified_variance_objective), 37
- assign_trimmed_neyman (assign_balance), 4
- binary_rectangle_corners, 5, 8–10, 12, 14, 15, 17, 18, 20, 21, 25, 28, 29, 31, 32, 34–38, 40
- binary_rectangle_regret
(binary_rectangle_corners), 5
- boundary_indicator
(coverage_indicator), 19
- boundary_rate (coverage_indicator), 19
- break_even_pilot_share, 3, 6, 23, 24

- certificate_valid (coverage_indicator), 19
- cmr_binary (cmr_two_arm), 15
- cmr_binary_from_rectangle (binary_rectangle_corners), 5
- cmr_delayed_outcome (cmr_proxy), 11
- cmr_multiarm, 6, 7, 9, 10, 12, 14, 15, 17, 18, 25
- cmr_multiarm_from_rectangle, 6, 8, 8, 10, 12, 14, 15, 17, 18, 20, 21, 25, 28, 29, 31, 32, 34–38, 40
- cmr_multiple_outcomes, 6, 8, 9, 9, 12, 14, 15, 17, 18, 25
- cmr_plan (pilot_plan), 22
- cmr_proxy, 6, 8–10, 11, 14, 15, 17, 18, 25
- cmr_stratified, 6, 8–10, 12, 12, 15, 17, 18, 25
- cmr_stratified_from_rectangle, 6, 8–10, 12, 14, 14, 17, 18, 20, 21, 25, 28, 29, 31, 32, 34–38, 40
- cmr_two_arm, 6, 8–10, 12, 14, 15, 15, 18, 25
- cmr_two_arm_from_rectangle (binary_rectangle_corners), 5
- cmr_unbounded, 6, 8–10, 12, 14, 15, 17, 17, 18, 25
- cmr_unbounded_from_rectangle, 6, 8–10, 12, 14, 15, 17, 18, 18, 20, 21, 25, 28, 29, 31, 32, 34–38, 40
- cmrdesign (cmrdesign-package), 2
- cmrdesign-package, 2
- coverage_indicator, 19
- folded_binomial_pmf, 6, 9, 15, 18, 20, 21, 28, 29, 31, 32, 34–38, 40
- folded_binomial_tails (folded_binomial_pmf), 20
- multiarm_oracle_variance (multiarm_variance_objective), 21
- multiarm_rectangle_vertices (multiarm_variance_objective), 21
- multiarm_regret (multiarm_variance_objective), 21
- multiarm_variance_objective, 5, 6, 9, 15, 18, 20, 21, 28, 29, 31, 32, 34–38, 40, 41
- oracle_variance (variance_objective), 41
- pilot_plan, 3, 7, 22, 24
- pilot_viability_band, 3, 7, 23, 23
- print.cmr_multiarm (print.cmr_two_arm), 24
- print.cmr_multiple_outcomes (print.cmr_two_arm), 24
- print.cmr_proxy (print.cmr_two_arm), 24
- print.cmr_stratified (print.cmr_two_arm), 24
- print.cmr_two_arm, 6, 8–10, 12, 14, 15, 17, 18, 24
- print.cmr_unbounded (print.cmr_two_arm), 24
- print.summary.cmr_result (print.cmr_two_arm), 24
- rectangle_bernoulli_binary, 6, 9, 15, 18, 20, 21, 26, 29, 31, 32, 34–38, 40
- rectangle_bernoulli_two_arm (rectangle_bernoulli_binary), 26
- rectangle_binary (rectangle_bernoulli_binary), 26
- rectangle_bounded_binary (rectangle_bounded_two_arm), 28
- rectangle_bounded_two_arm, 6, 9, 15, 18, 20, 21, 28, 28, 31, 32, 34–38, 40
- rectangle_delayed_outcome (rectangle_proxy), 32
- rectangle_multiarm, 6, 9, 15, 18, 20, 21, 28, 29, 30, 32, 34–38, 40
- rectangle_multiple_outcomes, 6, 9, 15, 18, 20, 21, 28, 29, 31, 31, 34–38, 40
- rectangle_proxy, 6, 9, 15, 18, 20, 21, 28, 29, 31, 32, 32, 35–38, 40
- rectangle_stratified, 6, 9, 15, 18, 20, 21, 28, 29, 31, 32, 34, 34, 36–38, 40
- rectangle_two_arm (rectangle_bernoulli_binary), 26
- rectangle_unbounded, 6, 9, 15, 18, 20, 21, 28, 29, 31, 32, 34, 35, 36, 37, 38, 40
- regret (variance_objective), 41
- saving_vs_balance (coverage_indicator), 19

share_of_oracle_gain
 (coverage_indicator), 19

stratified_oracle_variance
 (stratified_variance_objective),
 37

stratified_rectangle_vertices
 (stratified_variance_objective),
 37

stratified_regret
 (stratified_variance_objective),
 37

stratified_variance_objective, 5, 6, 9,
 15, 18, 20, 21, 28, 29, 31, 32, 34–36,
 37, 38, 40, 41

summary.cmr_multiarm
 (print.cmr_two_arm), 24

summary.cmr_multiple_outcomes
 (print.cmr_two_arm), 24

summary.cmr_proxy (print.cmr_two_arm),
 24

summary.cmr_stratified
 (print.cmr_two_arm), 24

summary.cmr_two_arm
 (print.cmr_two_arm), 24

summary.cmr_unbounded
 (print.cmr_two_arm), 24

variance_bounds_bernoulli_exact, 6, 9,
 15, 18, 20, 21, 28, 29, 31, 32, 34–37,
 38, 40

variance_bounds_martinez_taboada_ramdas
 (variance_bounds_maurer_pontil),
 39

variance_bounds_maurer_pontil, 6, 9, 15,
 18, 20, 21, 28, 29, 31, 32, 34–38, 39,
 40

variance_bounds_unbounded_mom, 6, 9, 15,
 18, 20, 21, 28, 29, 31, 32, 34–38, 40,
 40

variance_objective, 5, 21, 37, 41