Package: cosa (via r-universe)

September 6, 2024

Type Package

Title Bound Constrained Optimal Sample Size Allocation

Version 2.1.0

Date 2021-05-25

Description Implements bound constrained optimal sample size allocation (BCOSSA) framework described in Bulus & Dong (2021) <doi:10.1080/00220973.2019.1636197> for power analysis of multilevel regression discontinuity designs (MRDDs) and multilevel randomized trials (MRTs) with continuous outcomes. Minimum detectable effect size (MDES) and power computations for MRDDs allow polynomial functional form specification for the score variable (with or without interaction with the treatment indicator) (Bulus, in press).

Imports nloptr(>= 1.0.4), msm(>= 1.6.7)

Suggests knitr, rmarkdown

VignetteBuilder knitr

License GPL (>= 3)

NeedsCompilation no

Repository https://metinbulus.r-universe.dev

RemoteUrl https://github.com/metinbulus/cosa

RemoteRef HEAD

RemoteSha ff7ccbf1962b0171515712a417ba838f33c5065c

Contents

`cosa-package` *Bound Constrained Optimal Design of MRDDs and MRTs*

Description

Bound Constrained Optimal Sample Size Allocation (BCOSSA) functions are designed to optimize sample sizes at one or more levels subject to budget, statistical power, or effect size constraints. BCOSSA can be found in the following forms; (i) under budgetary constraints given marginal costs per unit while minimizing sampling variance of the treatment effect (or, alternatively, while maximizing power rate), (ii) under statistical power or effect size (ES) constraints while minimizing the total cost, and (iii) under sample size constraints for one or more levels along with (i) or (ii). Specifying `rhots = 0` or `order = 0` produces results equivalent to corresponding random assignment designs, which means there is no relationship between the treatment [random] and the score variable. Therefore, BCOSSA functions also allow optimization of treatment group sampling rate (`p`) under unequal marginal costs when primary constraint is placed on the total cost. Different starting values and algorithms may produce different results when marginal cost information is not provided and sample sizes at two or more levels and `p` are optimized. In such cases, experimenting different starting values and/or comparing several algorithms may faciliate decisions regarding sample sizes and `p`.

Designs available in the **cosa** package:

Design	*Total Levels*	*Treatment Level*	*Top Level*
`ird`	1	1	random
`bird2`	2	1	random
`bird2f1`	2	1	fixed
`bird3`	3	1	random
`bird4`	4	1	random
`crd2`	2	2	random
`bcrd3f2`	3	2	fixed
`bcrd3r2`	3	2	random
`bcrd4r2`	4	2	random
`crd3`	3	3	random
`bcrd4f3`	4	3	fixed
`bcrd4r3`	4	3	random

crd4	4	4	random

ird: individual-level regression discontinuity. bird: blocked individual-level regression discontinuity. crd: cluster-level regression discontinuity. bcrd: blocked cluster-level regression discontinuity.

Design parameters follow a sequential order. Numbers at the end of a sequential parameter refers to the corresponding level. For example rho2 is the proportion of variance in the outcome between level 2 units, rho3 is the proportion of variance in the outcome between level 3 units. Similiarly, r21 is the proportion of the variance in the outcome explained by level 1 covariates, r22 is the proportion of the variance in the outcome explained by level 2 covariates and so on. Similiar naming conventions applies to other design parameters.

bcrd3r2 *Blocked (Random) Cluster-level Regression Discontinuity (Three-level Design, Discontinuity at Level 2)*

Description

Use mdes.bcrd3r2() to calculate minimum detectable effect size, power.bcrd3r2() to calculate statistical power, and cosa.bcrd3r2() for bound constrained optimal sample size allocation (BCOSSA).

Usage

```
mdes.bcrd3r2(score = NULL, dists = "normal", k1 = -6, k2 = 6,
             order = 1, interaction = FALSE,
             treat.lower = TRUE, cutoff = 0, p = NULL,
             power = .80, alpha = .05, two.tailed = TRUE, df = n3 - g3 - 1,
             rho2, rho3, omega3, r21 = 0, r22 = 0, r2t3 = 0, g3 = 0,
             rate.tp = 1, rate.cc = 0, n1, n2, n3)

power.bcrd3r2(score = NULL, dists = "normal", k1 = -6, k2 = 6,
              order = 1, interaction = FALSE,
              treat.lower = TRUE, cutoff = 0, p = NULL,
              es = .25, alpha = .05, two.tailed = TRUE, df = n3 - g3 - 1,
              rho2, rho3, omega3, r21 = 0, r22 = 0, r2t3 = 0, g3 = 0,
              rate.tp = 1, rate.cc = 0, n1, n2, n3)

cosa.bcrd3r2(score = NULL, dists = "normal", k1 = -6, k2 = 6, rhots = NULL,
             order = 1, interaction = FALSE,
             treat.lower = TRUE, cutoff = 0, p = NULL,
             cn1 = 0, cn2 = 0, cn3 = 0, cost = NULL,
             n1 = NULL, n2 = NULL, n3 = NULL, n0 = c(10, 3, 100), p0 = .499,
             constrain = "power", round = TRUE, max.power = FALSE,
             local.solver = c("LBFGS", "SLSQP"),
             power = .80, es = .25, alpha = .05, two.tailed = TRUE,
             rho2, rho3, omega3, g3 = 0, r21 = 0, r22 = 0, r2t3 = 0)
```

Arguments

`score` vector or list; an empirical score variable or an object with class 'score' returned from the `inspect.score()` function.

`dists` character; distribution of the score variable, `"normal"` or `"uniform"`. By default, `dists = "normal"` specification implies a truncated normal distribution with `k1 = -6` and `k2 = 6`.

`k1` left truncation point for (uncentered) empirical, truncated normal, or uniform distribution. Ignored when `rhots = 0` or `order = 0`.

`k2` right truncation point for (uncentered) empirical, truncated normal, or uniform distribution. Ignored when `rhots = 0` or `order = 0`.

`order` integer >= 0; order of polynomial functional form specification for the score variable.

`interaction` logical; if `TRUE` polynomial specification interacts with the treatment variable.

`rhots` obsolote; use `order = 0` to obtain results equivalent to random assignment designs.

`treat.lower` logical; if `TRUE` units below the cutoff are treated.

`cutoff` decision threshold.

`p` proportion of level 2 units in the treatment condition.

`power` statistical power (1 - β).

`es` effect size (Cohen's d).

`alpha` probability of type I error (α).

`two.tailed` logical; `TRUE` for two-tailed hypothesis testing.

`df` degrees of freedom.

`rho2` proportion of variance in the outcome between level 2 units (unconditional ICC2).

`rho3` proportion of variance in the outcome between level 3 units (unconditional ICC3).

`omega3` ratio of the treatment effect variance between level 3 units to the variance in the outcome between level 3 units.

`g3` number of covariates at level 3.

`r21` proportion of level 1 variance in the outcome explained by level 1 covariates.

`r22` proportion of level 2 variance in the outcome explained by level 2 covariates.

`r2t3` proportion of treatment effect variance between level 3 units explained by level 3 covariates.

`rate.tp` treatment group participation rate.

`rate.cc` control group crossover rate.

`n1` average number of level 1 units per level 2 unit.

`n2` average number of level 2 units per level 3 unit.

`n3` number of level 3 units (blocks).

`cn1` marginal costs per level 1 unit in treatment and control conditions (positional), e.g. `c(10, 5)`.

`cn2` marginal costs per level 2 unit in treatment and control conditions (positional), e.g. `c(50, 20)`.

`cn3` marginal cost per level 3 unit.

`cost` total cost or budget. Ignored when `constrain = "power"` or `constrain = "es"`.

`p0` starting value for `p` when `rhots = 0` and `p = NULL`. Starting value is replaced with the average when `p` is constrained by bounds.

`n0` vector of starting values for `n1, n2, n3` (positional). Starting values are replaced with the averages when sample sizes are constrained by bounds.

`constrain` character; constrains one of the `"cost"`, `"power"`, or `"es"` at the specified value.

`round` logical; `TRUE` for rounded BCOSSA solution.

`max.power` logical; `TRUE` for maximizing the power rate instead of minimizing the variance. Applies when `constrain = "cost"`.

`local.solver` subset of `c("LBFGS", "SLSQP")`

Value

`parms` list of parameters used in the function.

`df` degrees of freedom.

`sse` standardized standard error.

`cosa` BCOSSA solution.

`mdes` minimum detectable effect size and (1 - α)% confidence limits.

`power` statistical power (1 - β)

Examples

```
score.obj <- inspect.score(rnorm(1000),
                           order = 1, interaction = FALSE,
                           cutoff = 0, k1 = -1, k2 = 1)

power.bcrd3r2(score.obj,
              es = 0.25, rho2 = .20, rho3 = .10, omega3 = .30,
              g3 = 0, r2t3 = 0, n1 = 50, n2 = 10, n3 = 10)

# minimum required number of level 2 units for each block
cosa.bcrd3r2(score.obj,
             es = 0.25, rho2 = .20, rho3 = .10, omega3 = .30,
             g3 = 0, r2t3 = 0,
             n1 = 50, n2 = NULL, n3 = 10)
```

bcrd4r2 *Blocked (Random) Cluster-level Regression Discontinuity (Four-level Design, Discontinuity at Level 2)*

Description

Use mdes.bcrd4r2() to calculate minimum detectable effect size, power.bcrd4r2() to calculate statistical power, and use cosa.bcrd4r2() for bound constrained optimal sample size allocation (BCOSSA).

Usage

```
mdes.bcrd4r2(score = NULL, dists = "normal", k1 = -6, k2 = 6,
             order = 1, interaction = FALSE,
             treat.lower = TRUE, cutoff = 0, p = NULL,
             power = .80, alpha = .05, two.tailed = TRUE, df = n4 - g4 - 1,
             rho2, rho3, rho4, omega3, omega4,
             r21 = 0, r22 = 0, r2t3 = 0, r2t4 = 0, g4 = 0,
             rate.tp = 1, rate.cc = 0, n1, n2, n3, n4)

power.bcrd4r2(score = NULL, dists = "normal", k1 = -6, k2 = 6,
              order = 1, interaction = FALSE,
              treat.lower = TRUE, cutoff = 0, p = NULL,
              es = .25, alpha = .05, two.tailed = TRUE, df = n4 - g4 - 1,
              rho2, rho3, rho4, omega3, omega4,
              r21 = 0, r22 = 0, r2t3 = 0, r2t4 = 0, g4 = 0,
              rate.tp = 1, rate.cc = 0, n1, n2, n3, n4)

cosa.bcrd4r2(score = NULL, dists = "normal", k1 = -6, k2 = 6, rhots = NULL,
             order = 1, interaction = FALSE,
             treat.lower = TRUE, cutoff = 0, p = NULL,
             cn1 = 0, cn2 = 0, cn3 = 0, cn4 = 0, cost = NULL,
             n1 = NULL, n2 = NULL, n3 = NULL, n4 = NULL,
             n0 = c(10, 3, 100, 5 + g4), p0 = .499,
             constrain = "power", round = TRUE, max.power = FALSE,
             local.solver = c("LBFGS", "SLSQP"),
             power = .80, es = .25, alpha = .05, two.tailed = TRUE,
             rho2, rho3, rho4, omega3, omega4,
             g4 = 0, r21 = 0, r22 = 0, r2t3 = 0, r2t4 = 0)
```

Arguments

score vector or list; an empirical score variable or an object with class 'score' returned from the inspect.score() function.

dists character; distribution of the score variable, "normal" or "uniform". By default, dists = "normal" specification implies a truncated normal distribution with k1 = -6 and k2 = 6.

`k1` left truncation point for (uncentered) empirical, truncated normal, or uniform distribution. Ignored when `rhots = 0` or `order = 0`.

`k2` right truncation point for (uncentered) empirical, truncated normal, or uniform distribution. Ignored when `rhots = 0` or `order = 0`.

`order` integer >= 0; order of polynomial functional form specification for the score variable.

`interaction` logical; if `TRUE` polynomial specification interacts with the treatment variable.

`rhots` obsolote; use `order = 0` to obtain results equivalent to random assignment designs.

`treat.lower` logical; if `TRUE` units below the cutoff are treated.

`cutoff` decision threshold.

`p` proportion of level 2 units in the treatment condition.

`power` statistical power (1 - β).

`es` effect size (Cohen's d).

`alpha` probability of type I error (α).

`two.tailed` logical; `TRUE` for two-tailed hypothesis testing.

`df` degrees of freedom.

`rho2` proportion of variance in the outcome between level 2 units (unconditional ICC2).

`rho3` proportion of variance in the outcome between level 3 units (unconditional ICC3).

`rho4` proportion of variance in the outcome between level 4 units (unconditional ICC4).

`omega3` ratio of the treatment effect variance between level 3 units to the variance in the outcome between level 3 units.

`omega4` ratio of the treatment effect variance between level 4 units to the variance in the outcome between level 4 units.

`g4` number of covariates at level 4.

`r21` proportion of level 1 variance in the outcome explained by level 1 covariates.

`r22` proportion of level 2 variance in the outcome explained by level 2 covariates.

`r2t3` proportion of treatment effect variance between level 3 units explained by level 3 covariates.

`r2t4` proportion of treatment effect variance between level 4 units explained by level 4 covariates.

`rate.tp` treatment group participation rate.

`rate.cc` control group crossover rate.

`n1` average number of level 1 units per level 2 unit.

`n2` average number of level 2 units per level 3 unit.

`n3` average number of level 3 units (blocks) per level 4 unit.

`n4` number of level 4 units (blocks).

`cn1` marginal costs per level 1 unit in treatment and control conditions (positional), e.g. `c(10, 5)`.

`cn2` marginal costs per level 2 unit in treatment and control conditions (positional), e.g. `c(50, 20)`.

`cn3` marginal cost per level 3 unit.

`cn4` marginal cost per level 4 unit.

`cost` total cost or budget. Ignored when `constrain = "power"` or `constrain = "es"`.

`p0` starting value for `p` when `rhots = 0` and `p = NULL`. Starting value is replaced with the average when `p` is constrained by bounds.

`n0` vector of starting values for `n1, n2, n3, n4` (positional). Starting values are replaced with the averages when sample sizes are constrained by bounds.

`constrain` character; constrains one of the `"cost"`, `"power"`, or `"es"` at the specified value.

`round` logical; `TRUE` for rounded BCOSSA solution.

`max.power` logical; `TRUE` for maximizing the power rate instead of minimizing the variance. Applies when `constrain = "cost"`.

`local.solver` subset of `c("LBFGS", "SLSQP")`.

Value

`parms` list of parameters used in the function.

`df` degrees of freedom.

`sse` standardized standard error.

`cosa` BCOSSA solution.

`mdes` minimum detectable effect size and (1 - α)% confidence limits.

`power` statistical power (1 - β)

Examples

```
score.obj <- inspect.score(rnorm(1000),
                           order = 1, interaction = FALSE,
                           cutoff = 0, k1 = -1, k2 = 1)

power.bcrd4r2(score.obj,
              es = 0.25, rho2 = .20, rho3 = .10, rho4 = .05,
              omega3 = .30, omega4 = .30,
              g4 = 0, r2t4 = 0,
              n1 = 20, n2 = 3, n3 = 20, n4 = 10)

# minimum required number of level 2 units for each one of the level 3 block
cosa.bcrd4r2(score.obj,
             es = 0.25, rho2 = .20, rho3 = .10, rho4 = .05,
             omega3 = .30, omega4 = .30,
             g4 = 0, r2t4 = 0,
             n1 = 20, n2 = NULL, n3 = 20, n4 = 10)
```

bcrd4r3 *Blocked (Random) Cluster-level Regression Discontinuity (Four-level Design, Discontinuity at Level 3)*

Description

Use `mdes.bcrd4r3()` to calculate minimum detectable effect size, `power.bcrd4r3()` to calculate statistical power, and `cosa.bcrd4r3()` for bound constrained optimal sample size allocation (BCOSSA).

Usage

```
mdes.bcrd4r3(score = NULL, dists = "normal", k1 = -6, k2 = 6,
             order = 1, interaction = FALSE,
             treat.lower = TRUE, cutoff = 0, p = NULL,
             power = .80, alpha = .05, two.tailed = TRUE, df = n4 - g4 - 1,
             rho2, rho3, rho4, omega4,
             r21 = 0, r22 = 0, r23 = 0, r2t4 = 0, g4 = 0,
             rate.tp = 1, rate.cc = 0, n1, n2, n3, n4)

power.bcrd4r3(score = NULL, dists = "normal", k1 = -6, k2 = 6,
              order = 1, interaction = FALSE,
              treat.lower = TRUE, cutoff = 0, p = NULL,
              es = .25, alpha = .05, two.tailed = TRUE, df = n4 - g4 - 1,
              rho2, rho3, rho4, omega4,
              r21 = 0, r22 = 0, r23 = 0, r2t4 = 0, g4 = 0,
              rate.tp = 1, rate.cc = 0, n1, n2, n3, n4)

cosa.bcrd4r3(score = NULL, dists = "normal", k1 = -6, k2 = 6, rhots = NULL,
             order = 1, interaction = FALSE,
             treat.lower = TRUE, cutoff = 0, p = NULL,
             cn1 = 0, cn2 = 0, cn3 = 0, cn4 = 0, cost = NULL,
             n1 = NULL, n2 = NULL, n3 = NULL, n4 = NULL,
             n0 = c(10, 3, 100, 5 + g4), p0 = .499,
             constrain = "power", round = TRUE, max.power = FALSE,
             local.solver = c("LBFGS", "SLSQP"),
             power = .80, es = .25, alpha = .05, two.tailed = TRUE,
             rho2, rho3, rho4, omega4,
             g4 = 0, r21 = 0, r22 = 0, r23 = 0, r2t4 = 0)
```

Arguments

`score` vector or list; an empirical score variable or an object with class ’score’ returned from the `inspect.score()` function.

`dists` character; distribution of the score variable, `"normal"` or `"uniform"`. By default, `dists = "normal"` specification implies a truncated normal distribution with `k1 = -6` and `k2 = 6`.

`k1` left truncation point for (uncentered) empirical, truncated normal, or uniform distribution. Ignored when `rhots = 0` or `order = 0`.

`k2` right truncation point for (uncentered) empirical, truncated normal, or uniform distribution. Ignored when `rhots = 0` or `order = 0`.

`order` integer >= 0; order of polynomial functional form specification for the score variable.

`interaction` logical; if `TRUE` polynomial specification interacts with the treatment variable.

`rhots` obsolote; use `order = 0` to obtain results equivalent to random assignment designs.

`treat.lower` logical; if `TRUE` units below the cutoff are treated.

`cutoff` decision threshold.

`p` proportion of level 3 units in the treatment condition.

`power` statistical power (1 - β).

`es` effect size (Cohen's d).

`alpha` probability of type I error (α).

`two.tailed` logical; `TRUE` for two-tailed hypothesis testing.

`df` degrees of freedom.

`rho2` proportion of variance in the outcome between level 2 units (unconditional ICC2).

`rho3` proportion of variance in the outcome between level 3 units (unconditional ICC3).

`rho4` proportion of variance in the outcome between level 4 units (unconditional ICC4).

`omega4` ratio of the treatment effect variance between level 4 units to the variance in the outcome between level 4 units.

`g4` number of covariates at level 4.

`r21` proportion of level 1 variance in the outcome explained by level 1 covariates.

`r22` proportion of level 2 variance in the outcome explained by level 2 covariates.

`r23` proportion of level 3 variance in the outcome explained by level 3 covariates.

`r2t4` proportion of treatment effect variance between level 4 units explained by level 4 covariates.

`rate.tp` treatment group participation rate.

`rate.cc` control group crossover rate.

`n1` average number of level 1 units per level 2 unit.

`n2` average number of level 2 units per level 3 unit.

`n3` average number of level 3 units per level 4 unit.

`n4` number of level 4 units (blocks).

`cn1` marginal costs per level 1 unit in treatment and control conditions (positional), e.g. `c(10, 5)`.

`cn2` marginal costs per level 2 unit in treatment and control conditions (positional), e.g. `c(50, 20)`.

`cn3` marginal costs per level 3 unit in treatment and control conditions (positional), e.g. `c(80, 50)`.

`cn4` marginal cost per level 4 unit.

`cost` total cost or budget. Ignored when `constrain = "power"` or `constrain = "es"`.

`p0` starting value for `p` when `rhots = 0` and `p = NULL`. Starting value is replaced with the average when `p` is constrained by bounds.

`n0` vector of starting values for `n1, n2, n3, n4` (positional). Starting values are replaced with the averages when sample sizes are constrained by bounds.

`constrain` character; constrains one of the `"cost"`, `"power"`, or `"es"` at the specified value.

`round` logical; `TRUE` for rounded BCOSSA solution.

`max.power` logical; `TRUE` for maximizing the power rate instead of minimizing the variance. Applies when `constrain = "cost"`.

`local.solver` subset of `c("LBFGS", "SLSQP")`.

Value

`parms` list of parameters used in the function.

`df` degrees of freedom.

`sse` standardized standard error.

`cosa` BCOSSA solution.

`mdes` minimum detectable effect size and (1 - α)% confidence limits.

`power` statistical power (1 - β)

Examples

```
score.obj <- inspect.score(rnorm(1000),
                           order = 1, interaction = FALSE,
                           cutoff = 0, k1 = -1, k2 = 1)

power.bcrd4r3(score.obj,
              es = 0.25, rho2 = .20, rho3 = .10, rho4 = .05,
              omega4 = .30, g4 = 0, r2t4 = 0,
              n1 = 20, n2 = 3, n3 = 20, n4 = 10)

# minimum required number of level 3 units for each one of the level 4 block
cosa.bcrd4r3(score.obj,
             es = 0.25, rho2 = .20, rho3 = .10, rho4 = .05,
             omega4 = .30, g4 = 0, r2t4 = 0,
             n1 = 20, n2 = 3, n3 = NULL, n4 = 10)
```

bird2 *Blocked (Random) Individual-level Regression Discontinuity (Two-level Design, Discontinuity at Level 1)*

Description

Use `mdes.bird2()` to calculate minimum detectable effect size, `power.bird2()` to calculate statistical power, and `cosa.bird2()` for bound constrained optimal sample size allocation (BCOSSA).

Usage

```
mdes.bird2(score = NULL, dists = "normal", k1 = -6, k2 = 6,
           order = 1, interaction = FALSE,
           treat.lower = TRUE, cutoff = 0, p = NULL,
           power = .80, alpha = .05, two.tailed = TRUE, df = n2 - g2 - 1,
           rho2, omega2, r21 = 0, r2t2 = 0, g2 = 0,
           rate.tp = 1, rate.cc = 0, n1, n2)

power.bird2(score = NULL, dists = "normal", k1 = -6, k2 = 6,
            order = 1, interaction = FALSE,
            treat.lower = TRUE, cutoff = 0, p = NULL,
            es = .25, alpha = .05, two.tailed = TRUE, df = n2 - g2 - 1,
            rho2, omega2, r21 = 0, r2t2 = 0, g2 = 0,
            rate.tp = 1, rate.cc = 0, n1, n2)

cosa.bird2(score = NULL, dists = "normal", k1 = -6, k2 = 6, rhots = NULL,
           order = 1, interaction = FALSE,
           treat.lower = TRUE, cutoff = 0, p = NULL,
           cn1 = 0, cn2 = 0, cost = NULL,
           n1 = NULL, n2 = NULL, n0 = c(10, 100), p0 = .499,
           constrain = "power", round = TRUE, max.power = FALSE,
           local.solver = c("LBFGS", "SLSQP"),
           power = .80, es = .25, alpha = .05, two.tailed = TRUE,
           rho2, omega2, g2 = 0, r21 = 0, r2t2 = 0)
```

Arguments

`score`	vector or list; an empirical score variable or an object with class 'score' returned from the `inspect.score()` function.
`dists`	character; distribution of the score variable, `"normal"` or `"uniform"`. By default, `dists = "normal"` specification implies a truncated normal distribution with `k1 = -6` and `k2 = 6`.
`k1`	left truncation point for (uncentered) empirical, truncated normal, or uniform distribution. Ignored when `rhots = 0` or `order = 0`.
`k2`	right truncation point for (uncentered) empirical, truncated normal, or uniform distribution. Ignored when `rhots = 0` or `order = 0`.

`order` integer >= 0; order of polynomial functional form specification for the score variable.

`interaction` logical; if `TRUE` polynomial specification interacts with the treatment variable.

`rhots` obsolote; use `order = 0` to obtain results equivalent to random assignment designs.

`treat.lower` logical; if `TRUE` units below the cutoff are treated.

`cutoff` decision threshold.

`p` proportion of level 1 units in the treatment condition.

`power` statistical power (1 - β).

`es` effect size (Cohen's d).

`alpha` probability of type I error (α).

`two.tailed` logical; `TRUE` for two-tailed hypothesis testing.

`df` degrees of freedom.

`rho2` proportion of variance in the outcome between level 2 units (unconditional ICC2).

`omega2` ratio of the treatment effect variance between level 2 units to the variance in the outcome between level 2 units.

`g2` number of covariates at level 2.

`r21` proportion of level 1 variance in the outcome explained by level 1 covariates.

`r2t2` proportion of treatment effect variance between level 2 units explained by level 2 covariates.

`rate.tp` treatment group participation rate.

`rate.cc` control group crossover rate.

`n1` average number of level 1 units per level 2 units.

`n2` number of level 2 units (blocks).

`cn1` marginal costs per level 1 unit in treatment and control conditions (positional), e.g. `c(10, 5)`.

`cn2` marginal cost per level 2 unit.

`cost` total cost or budget. Ignored when `constrain = "power"` or `constrain = "es"`.

`p0` starting value for `p` when `rhots = 0` and `p = NULL`. Starting value is replaced with the average when `p` is constrained by bounds.

`n0` vector of starting values for `n1, n2` (positional). Starting values are replaced with the averages when sample sizes are constrained by bounds.

`constrain` character; constrains one of the `"cost"`, `"power"`, or `"es"` at the specified value.

`round` logical; `TRUE` for rounded BCOSSA solution.

`max.power` logical; `TRUE` for maximizing the power rate instead of minimizing the variance. Applies when `constrain = "cost"`.

`local.solver` subset of `c("LBFGS", "SLSQP")`

Value

parms	list of parameters used in the function.
df	degrees of freedom.
sse	standardized standard error.
cosa	BCOSSA solution.
mdes	minimum detectable effect size and (1 - α)% confidence limits.
power	statistical power (1 - β)

Examples

```
score.obj <- inspect.score(rnorm(1000),
                            order = 1, interaction = FALSE,
                            cutoff = 0, k1 = -1, k2 = 1)

power.bird2(score.obj,
            es = 0.25, rho2 = .20, omega2 = .30,
            g2 = 0, r2t2 = 0, n1 = 50, n2 = 30)

# minimum required number of level 1 units for each one of the level 2 block
cosa.bird2(score.obj,
           es = 0.25, rho2 = .20, omega2 = .30,
           g2 = 0, r2t2 = 0,
           n1 = NULL, n2 = 30)
```

bird3 *Blocked (Random) Individual-level Regression Discontinuity (Three-level Design, Discontinuity at Level 1)*

Description

Use mdes.bird3() to calculate minimum detectable effect size, power.bird3() to calculate statistical power, and cosa.bird3() for bound constrained optimal sample size allocation (BCOSSA).

Usage

```
mdes.bird3(score = NULL, dists = "normal", k1 = -6, k2 = 6,
           order = 1, interaction = FALSE,
           treat.lower = TRUE, cutoff = 0, p = NULL,
           power = .80, alpha = .05, two.tailed = TRUE, df = n3 - g3 - 1,
           rho2, rho3, omega2, omega3, r21 = 0, r2t2 = 0, r2t3 = 0, g3 = 0,
           rate.tp = 1, rate.cc = 0, n1, n2, n3)

power.bird3(score = NULL, dists = "normal", k1 = -6, k2 = 6,
            order = 1, interaction = FALSE,
            treat.lower = TRUE, cutoff = 0, p = NULL,
            es = .25, alpha = .05, two.tailed = TRUE, df = n3 - g3 - 1,
             rho2, rho3, omega2, omega3, r21 = 0, r2t2 = 0, r2t3 = 0, g3 = 0,
             rate.tp = 1, rate.cc = 0, n1, n2, n3)

cosa.bird3(score = NULL, dists = "normal", k1 = -6, k2 = 6, rhots = NULL,
           order = 1, interaction = FALSE,
           treat.lower = TRUE, cutoff = 0, p = NULL,
           cn1 = 0, cn2 = 0, cn3 = 0, cost = NULL,
           n1 = NULL, n2 = NULL, n3 = NULL,
           n0 = c(10, 3, 100), p0 = .499,
           constrain = "power", round = TRUE, max.power = FALSE,
           local.solver = c("LBFGS", "SLSQP"),
           power = .80, es = .25, alpha = .05, two.tailed = TRUE,
           rho2, rho3, omega2, omega3,
           g3 = 0, r21 = 0, r2t2 = 0, r2t3 = 0)
```

Arguments

`score` vector or list; an empirical score variable or an object with class 'score' returned from the `inspect.score()` function.

`dists` character; distribution of the score variable, `"normal"` or `"uniform"`. By default, `dists = "normal"` specification implies a truncated normal distribution with `k1 = -6` and `k2 = 6`.

`k1` left truncation point for (uncentered) empirical, truncated normal, or uniform distribution. Ignored when `rhots = 0` or `order = 0`.

`k2` right truncation point for (uncentered) empirical, truncated normal, or uniform distribution. Ignored when `rhots = 0` or `order = 0`.

`order` integer >= 0; order of polynomial functional form specification for the score variable.

`interaction` logical; if `TRUE` polynomial specification interacts with the treatment variable.

`rhots` obsolote; use `order = 0` to obtain results equivalent to random assignment designs.

`treat.lower` logical; if `TRUE` units below the cutoff are treated.

`cutoff` decision threshold.

`p` proportion of level 1 units in the treatment condition.

`power` statistical power (1 - β).

`es` effect size (Cohen's d).

`alpha` probability of type I error (α).

`two.tailed` logical; `TRUE` for two-tailed hypothesis testing.

`df` degrees of freedom.

`rho2` proportion of variance in the outcome between level 2 units (unconditional ICC2).

`rho3` proportion of variance in the outcome between level 3 units (unconditional ICC3).

`omega2` ratio of the treatment effect variance between level 2 units to the variance in the outcome between level 2 units.

`omega3` ratio of the treatment effect variance between level 3 units to the variance in the outcome between level 3 units.

`g3` number of covariates at level 3.

`r21` proportion of level 1 variance in the outcome explained by level 1 covariates.

`r2t2` proportion of treatment effect variance between level 2 units explained by level 2 covariates.

`r2t3` proportion of treatment effect variance between level 3 units explained by level 3 covariates.

`rate.tp` treatment group participation rate.

`rate.cc` control group crossover rate.

`n1` average number of level 1 units per level 2 unit.

`n2` average number of level 2 units (blocks) per level 3 unit.

`n3` number of level 3 units (blocks).

`cn1` marginal costs per level 1 unit in treatment and control conditions (positional), e.g. `c(10, 5)`.

`cn2` marginal cost per level 2 unit.

`cn3` marginal cost per level 3 unit.

`cost` total cost or budget. Ignored when `constrain = "power"` or `constrain = "es"`.

`p0` starting value for `p` when `rhots = 0` and `p = NULL`. Starting value is replaced with the average when `p` is constrained by bounds.

`n0` vector of starting values for `n1, n2, n3` (positional). Starting values are replaced with the averages when sample sizes are constrained by bounds.

`constrain` character; constrains one of the `"cost"`, `"power"`, or `"es"` at the specified value.

`round` logical; `TRUE` for rounded BCOSSA solution.

`max.power` logical; `TRUE` for maximizing the power rate instead of minimizing the variance. Applies when `constrain = "cost"`.

`local.solver` subset of `c("LBFGS", "SLSQP")`

Value

`parms` list of parameters used in the function.

`df` degrees of freedom.

`sse` standardized standard error.

`cosa` BCOSSA solution.

`mdes` minimum detectable effect size and (1 - α)% confidence limits.

`power` statistical power (1 - β)

Examples

```
score.obj <- inspect.score(rnorm(1000),
                           order = 1, interaction = FALSE,
                           cutoff = 0, k1 = -1, k2 = 1)

power.bird3(score.obj,
            es = 0.25, rho2 = .20, rho3 = .10,
            omega2 = .30, omega3 = .30,
            g3 = 0, r2t3 = 0,
            n1 = 20, n2 = 3, n3 = 20)

# minimum required number of level 1 units for each one of the level 2 block
cosa.bird3(score.obj,
           es = 0.25, rho2 = .20, rho3 = .10,
           omega2 = .30, omega3 = .30,
           g3 = 0, r2t3 = 0,
           n1 = NULL, n2 = 3, n3 = 20)
```

---

bird4 — *Blocked (Random) Individual-level Regression Discontinuity (Four-level Design, Discontinuity at Level 1)*

Description

Use `mdes.bird4()` to calculate minimum detectable effect size, `power.bird4()` to calculate statistical power, and `cosa.bird4()` for bound constrained optimal sample size allocation (BCOSSA).

Usage

```
mdes.bird4(score = NULL, dists = "normal", k1 = -6, k2 = 6,
           order = 1, interaction = FALSE,
           treat.lower = TRUE, cutoff = 0, p = NULL,
           power = .80, alpha = .05, two.tailed = TRUE, df = n4 - g4 - 1,
           rho2, rho3, rho4, omega2, omega3, omega4,
           r21 = 0, r2t2 = 0, r2t3 = 0, r2t4 = 0, g4 = 0,
           rate.tp = 1, rate.cc = 0, n1, n2, n3, n4)

power.bird4(score = NULL, dists = "normal", k1 = -6, k2 = 6,
            order = 1, interaction = FALSE,
            treat.lower = TRUE, cutoff = 0, p = NULL,
            es = .25, alpha = .05, two.tailed = TRUE, df = n4 - g4 - 1,
            rho2, rho3, rho4, omega2, omega3, omega4,
            r21 = 0, r2t2 = 0, r2t3 = 0, r2t4 = 0, g4 = 0,
            rate.tp = 1, rate.cc = 0, n1, n2, n3, n4)

cosa.bird4(score = NULL, dists = "normal", k1 = -6, k2 = 6, rhots = NULL,
           order = 1, interaction = FALSE,
           treat.lower = TRUE, cutoff = 0, p = NULL,
           cn1 = 0, cn2 = 0, cn3 = 0, cn4 = 0, cost = NULL,
           n1 = NULL, n2 = NULL, n3 = NULL, n4 = NULL,
           n0 = c(10, 3, 100, 5 + g4), p0 = .499,
           constrain = "power", round = TRUE, max.power = FALSE,
           local.solver = c("LBFGS", "SLSQP"),
           power = .80, es = .25, alpha = .05, two.tailed = TRUE,
           rho2, rho3, rho4, omega2, omega3, omega4,
           g4 = 0, r21 = 0, r2t2 = 0, r2t3 = 0, r2t4 = 0)
```

Arguments

`score` vector or list; an empirical score variable or an object with class 'score' returned from the `inspect.score()` function.

`dists` character; distribution of the score variable, `"normal"` or `"uniform"`. By default, `dists = "normal"` specification implies a truncated normal distribution with `k1 = -6` and `k2 = 6`.

`k1` left truncation point for (uncentered) empirical, truncated normal, or uniform distribution. Ignored when `rhots = 0` or `order = 0`.

`k2` right truncation point for (uncentered) empirical, truncated normal, or uniform distribution. Ignored when `rhots = 0` or `order = 0`.

`order` integer >= 0; order of polynomial functional form specification for the score variable.

`interaction` logical; if `TRUE` polynomial specification interacts with the treatment variable.

`rhots` obsolote; use `order = 0` to obtain results equivalent to random assignment designs.

`treat.lower` logical; if `TRUE` units below the cutoff are treated.

`cutoff` decision threshold.

`p` proportion of level 1 units in the treatment condition.

`power` statistical power (1 - β).

`es` effect size (Cohen's d).

`alpha` probability of type I error (α).

`two.tailed` logical; `TRUE` for two-tailed hypothesis testing.

`df` degrees of freedom.

`rho2` proportion of variance in the outcome between level 2 units (unconditional ICC2).

`rho3` proportion of variance in the outcome between level 3 units (unconditional ICC3).

`rho4` proportion of variance in the outcome between level 4 units (unconditional ICC4).

`omega2` ratio of the treatment effect variance between level 2 units to the variance in the outcome between level 2 units.

`omega3` ratio of the treatment effect variance between level 3 units to the variance in the outcome between level 3 units.

`omega4` ratio of the treatment effect variance between level 4 units to the variance in the outcome between level 4 units.

`g4` number of covariates at level 4.

`r21` proportion of level 1 variance in the outcome explained by level 1 covariates.

`r2t2` proportion of treatment effect variance between level 2 units explained by level 2 covariates.

`r2t3` proportion of treatment effect variance between level 3 units explained by level 3 covariates.

`r2t4` proportion of treatment effect variance between level 4 units explained by level 4 covariates.

`rate.tp` treatment group participation rate.

`rate.cc` control group crossover rate.

`n1` average number of level 1 units per level 2 unit.

`n2` average number of level 2 units (blocks) per level 3 unit.

`n3` average number of level 3 units (blcoks) per level 4 unit.

`n4` number of level 4 units (blocks).

`cn1` marginal costs per level 1 unit in treatment and control conditions (positional), e.g. `c(10, 5)`.

`cn2` marginal cost per level 2 unit.

`cn3` marginal cost per level 3 unit.

`cn4` marginal cost per level 4 unit.

`cost` total cost or budget. Ignored when `constrain = "power"` or `constrain = "es"`.

`p0` starting value for `p` when `rhots = 0` and `p = NULL`. Starting value is replaced with the average when `p` is constrained by bounds.

`n0` vector of starting values for `n1, n2, n3, n4` (positional). Starting values are replaced with the averages when sample sizes are constrained by bounds.

`constrain` character; constrains one of the `"cost"`, `"power"`, or `"es"` at the specified value.

`round` logical; `TRUE` for rounded BCOSSA solution.

`max.power` logical; `TRUE` for maximizing the power rate instead of minimizing the variance. Applies when `constrain = "cost"`.

`local.solver` subset of `c("LBFGS", "SLSQP")`.

Value

`parms` list of parameters used in the function.

`df` degrees of freedom.

`sse` standardized standard error.

`cosa` BCOSSA solution.

`mdes` minimum detectable effect size and (1 - α)% confidence limits.

`power` statistical power (1 - β)

Examples

```
score.obj <- inspect.score(rnorm(1000),
                           order = 1, interaction = FALSE,
                           cutoff = 0, k1 = -1, k2 = 1)

power.bird4(score.obj,
            es = .25, rho2 = .20, rho3 = .10, rho4 = .05,
            omega2 = .30, omega3 = .30, omega4 = .30,
            g4 = 0, r2t4 = 0, n1 = 20, n2 = 3, n3 = 20, n4 = 5)

# minimum required number of level 1 units for each one of the level 2 block
cosa.bird4(score.obj, order = 2,
           es = .25, rho2 = .20, rho3 = .10, rho4 = .05,
           omega2 = .30, omega3 = .30, omega4 = .30,
           g4 = 0, r2t4 = 0, n1 = NULL, n2 = 3, n3 = 20, n4 = 5)
```

`cosa-deprecated` *Deprecated and Defunct functions in* **cosa**

Description

Some function are renamed and depreciated. They may be removed in the future.

Details

Depreciated function names:

- `power.crd2r2` is depreciated, use `power.crd2` instead.
- `mdes.crd2r2` is depreciated, use `mdes.crd2` instead.
- `cosa.crd2r2` is depreciated, use `cosa.crd2` instead.
- `power.crd3r3` is depreciated, use `power.crd3` instead.
- `mdes.crd3r3` is depreciated, use `mdes.crd3` instead.
- `cosa.crd3r3` is depreciated, use `cosa.crd3` instead.
- `power.crd4r4` is depreciated, use `power.crd4` instead.
- `mdes.crd4r4` is depreciated, use `mdes.crd4` instead.
- `cosa.crd4r4` is depreciated, use `cosa.crd4` instead.
- `power.ira1r1` is depreciated, use `power.ira` instead.
- `mdes.ira1r1` is depreciated, use `mdes.ira` instead.
- `power.bira2r1` is depreciated, use `power.bira2` instead.
- `mdes.bira2r1` is depreciated, use `mdes.bira2` instead.
- `cosa.bira2r1` is depreciated, use `cosa.bira2` instead.
- `power.bira3r1` is depreciated, use `power.bira3` instead.
- `mdes.bira3r1` is depreciated, use `mdes.bira3` instead.

- `cosa.bira3r1` is depreciated, use `cosa.bira3` instead.
- `power.bira4r1` is depreciated, use `power.bira4` instead.
- `mdes.bira4r1` is depreciated, use `mdes.bira4` instead.
- `cosa.bira4r1` is depreciated, use `cosa.bira4` instead.

crd2 — *Cluster-level Regression Discontinuity (Two-level Design, Discontinuity at Level 2, w/ or w/o Strata or Fixed Blocks)*

Description

Use `mdes.crd2()` to calculate minimum detectable effect size, `power.crd2()` to calculate statistical power, and `cosa.crd2()` for bound constrained optimal sample size allocation (BCOSSA). If higher level strata or fixed blocks exist, use `mdes.bcrd3f2()` to calculate minimum detectable effect size, `power.bcrd3f2()` to calculate statistical power, and `cosa.bcrd3f2()` for BCOSSA.

Usage

```
mdes.crd2(score = NULL, dists = "normal", k1 = -6, k2 = 6,
          order = 1, interaction = FALSE,
          treat.lower = TRUE, cutoff = 0, p = NULL,
          power = .80, alpha = .05, two.tailed = TRUE,
          df = n2 - g2 - order * (1 + interaction) - 2,
          rho2, r21 = 0, r22 = 0, g2 = 0, rate.tp = 1, rate.cc = 0, n1, n2)

power.crd2(score = NULL, dists = "normal", k1 = -6, k2 = 6,
           order = 1, interaction = FALSE,
           treat.lower = TRUE, cutoff = 0, p = NULL,
           es = .25, alpha = .05, two.tailed = TRUE,
           df = n2 - g2 - order * (1 + interaction) - 2,
           rho2, r21 = 0, r22 = 0, g2 = 0, rate.tp = 1, rate.cc = 0, n1, n2)

cosa.crd2(score = NULL, dists = "normal", k1 = -6, k2 = 6, rhots = NULL,
          order = 1, interaction = FALSE,
          treat.lower = TRUE, cutoff = 0, p = NULL,
          cn1 = 0, cn2 = 0, cost = NULL,
          n1 = NULL, n2 = NULL, n0 = c(10, 100), p0 = .499,
          constrain = "power", round = TRUE,
          max.power = FALSE, local.solver = c("LBFGS", "SLSQP"),
          power = .80, es = .25, alpha = .05, two.tailed = TRUE,
          rho2, g2 = 0, r21 = 0, r22 = 0)

mdes.bcrd3f2(score = NULL, dists = "normal", k1 = -6, k2 = 6,
             order = 1, interaction = FALSE,
             treat.lower = TRUE, cutoff = 0, p = NULL,
             power = .80, alpha = .05, two.tailed = TRUE,
             df = n3 * (n2 - 2) - g2 - order * (1 + interaction),
             rho2, r21 = 0, r22 = 0, g2 = 0,
             rate.tp = 1, rate.cc = 0, n1, n2, n3)

power.bcrd3f2(score = NULL, dists = "normal", k1 = -6, k2 = 6,
              order = 1, interaction = FALSE,
              treat.lower = TRUE, cutoff = 0, p = NULL,
              es = .25, alpha = .05, two.tailed = TRUE,
              df = n3 * (n2 - 2) - g2 - order * (1 + interaction),
              rho2, r21 = 0, r22 = 0, g2 = 0,
              rate.tp = 1, rate.cc = 0, n1, n2, n3)

cosa.bcrd3f2(score = NULL, dists = "normal", k1 = -6, k2 = 6, rhots = NULL,
             order = 1, interaction = FALSE,
             treat.lower = TRUE, cutoff = 0, p = NULL,
             cn1 = 0, cn2 = 0, cn3 = 0, cost = NULL,
             n1 = NULL, n2 = NULL, n3 = NULL,
             n0 = c(10, 100, 5), p0 = .499,
             constrain = "power", round = TRUE, max.power = FALSE,
             local.solver = c("LBFGS", "SLSQP"),
             power = .80, es = .25, alpha = .05, two.tailed = TRUE,
             rho2, g2 = 0, r21 = 0, r22 = 0)
```

Arguments

`score`	vector or list; an empirical score variable or an object with class 'score' returned from the `inspect.score()` function.
`dists`	character; distribution of the score variable, `"normal"` or `"uniform"`. By default, `dists = "normal"` specification implies a truncated normal distribution with `k1 = -6` and `k2 = 6`.
`k1`	left truncation point for (uncentered) empirical, truncated normal, or uniform distribution. Ignored when `rhots = 0` or `order = 0`.
`k2`	right truncation point for (uncentered) empirical, truncated normal, or uniform distribution. Ignored when `rhots = 0` or `order = 0`.
`order`	integer >= 0; order of polynomial functional form specification for the score variable.
`interaction`	logical; if `TRUE` polynomial specification interacts with the treatment variable.
`rhots`	obsolote; use `order = 0` to obtain results equivalent to random assignment designs.
`treat.lower`	logical; if `TRUE` units below the cutoff are treated.
`cutoff`	decision threshold.
`p`	proportion of level 2 units in the treatment condition.
`power`	statistical power (1 - β).
`es`	effect size (Cohen's d).
`alpha`	probability of type I error (α).

two.tailed logical; TRUE for two-tailed hypothesis testing.

df degrees of freedom.

rho2 proportion of variance in the outcome between level 2 units (unconditional ICC2).

g2 number of covariates at level 2.

r21 proportion of level 1 variance in the outcome explained by level 1 covariates.

r22 proportion of level 2 variance in the outcome explained by level 2 covariates.

rate.tp treatment group participation rate.

rate.cc control group crossover rate.

n1 average number of level 1 units per level 2 unit.

n2 number of level 2 units (per stratum or block, if exists).

n3 number of stratum or fixed blocks.

cn1 marginal costs per level 1 unit in treatment and control conditions (positional), e.g. c(10, 5).

cn2 marginal costs per level 2 unit in treatment and control conditions (positional), e.g. c(50, 30).

cn3 marginal cost per stratum or fixed block.

cost total cost or budget. Ignored when constrain = "power" or constrain = "es".

n0 vector of starting values for n1, n2 or n1, n2, n3 (positional). Starting values are replaced with the averages when sample sizes are constrained by bounds.

p0 starting value for p when rhots = 0 or order = 0, and p = NULL. Starting value is replaced with the average when p is constrained by bounds.

constrain character; constrains one of the "cost", "power", or "es" at the specified value.

round logical; TRUE for rounded BCOSSA solution.

max.power logical; TRUE for maximizing the power rate instead of minimizing the variance. Applies when constrain = "cost".

local.solver subset of c("LBFGS", "SLSQP")

Value

parms list of parameters used in the function.

df degrees of freedom.

sse standardized standard error.

cosa BCOSSA solution.

mdes minimum detectable effect size and (1 - α)% confidence limits.

power statistical power (1 - β)

Examples

```
score.obj <- inspect.score(rnorm(1000),
                           order = 1, interaction = FALSE,
                           cutoff = 0, k1 = -1, k2 = 1)

# single site (no blocks)
power.crd2(score.obj,
           es = .25, rho2 = .20, g2 = 0, r22 = 0,
           n1 = 50, n2 = 30)

# with 5 blocks (note that r22 is modified but g2 remains the same)
power.bcrd3f2(score.obj,
              es = .25, rho2 = .20, g2 = 0, r22 = .30,
              n1 = 50, n2 = 30, n3 = 5)

# minimum required number of level 2 units for each block
cosa.bcrd3f2(score.obj,
             es = .25, rho2 = .20, g2 = 0, r22 = .30,
             n1 = 50, n2 = NULL, n3 = 5)
```

crd3 — *Cluster-level Regression Discontinuity (Three-level Design, Discontinuity at Level 3, w/ or w/o Strata or Fixed Blocks)*

Description

Use `mdes.crd3()` to calculate minimum detectable effect size, `power.crd3()` to calculate statistical power, and `cosa.crd3()` for bound constrained optimal sample size allocation (BCOSSA). If higher level strata or fixed blocks exist, use `mdes.bcrd4f3()` to calculate minimum detectable effect size, `power.bcrd4f3()` to calculate statistical power, and `cosa.bcrd4f3()` for BCOSSA.

Usage

```
mdes.crd3(score = NULL, dists = "normal", k1 = -6, k2 = 6,
          order = 1, interaction = FALSE,
          treat.lower = TRUE, cutoff = 0, p = NULL,
          power = .80, alpha = .05, two.tailed = TRUE,
          df = n3 - g3 - order * (1 + interaction) - 2,
          rho2, rho3, r21 = 0, r22 = 0, r23 = 0,
          g3 = 0, rate.tp = 1, rate.cc = 0, n1, n2, n3)

power.crd3(score = NULL, dists = "normal", k1 = -6, k2 = 6,
           order = 1, interaction = FALSE,
           treat.lower = TRUE, cutoff = 0, p = NULL,
           es = .25, alpha = .05, two.tailed = TRUE,
           df = n3 - g3 - order * (1 + interaction) - 2,
           rho2, rho3, r21 = 0, r22 = 0, r23 = 0,
           g3 = 0, rate.tp = 1, rate.cc = 0, n1, n2, n3)

cosa.crd3(score = NULL, dists = "normal", k1 = -6, k2 = 6, rhots = NULL,
          order = 1, interaction = FALSE,
          treat.lower = TRUE, cutoff = 0, p = NULL,
          cn1 = 0, cn2 = 0, cn3 = 0, cost = NULL,
          n1 = NULL, n2 = NULL, n3 = NULL,
          n0 = c(10, 3, 100), p0 = .499,
          constrain = "power", round = TRUE, max.power = FALSE,
          local.solver = c("LBFGS", "SLSQP"),
          power = .80, es = .25, alpha = .05, two.tailed = TRUE,
          rho2, rho3, g3 = 0, r21 = 0, r22 = 0, r23 = 0)

mdes.bcrd4f3(score = NULL, dists = "normal", k1 = -6, k2 = 6,
             order = 1, interaction = FALSE,
             treat.lower = TRUE, cutoff = 0, p = NULL,
             power = .80, alpha = .05, two.tailed = TRUE,
             df = n4 * (n3 - 2) - g3 - order * (1 + interaction),
             rho2, rho3, r21 = 0, r22 = 0, r23 = 0, g3 = 0,
             rate.tp = 1, rate.cc = 0, n1, n2, n3, n4)

power.bcrd4f3(score = NULL, dists = "normal", k1 = -6, k2 = 6,
              order = 1, interaction = FALSE,
              treat.lower = TRUE, cutoff = 0, p = NULL,
              es = .25, alpha = .05, two.tailed = TRUE,
              df = n4 * (n3 - 2) - g3 - order * (1 + interaction),
              rho2, rho3, r21 = 0, r22 = 0, r23 = 0, g3 = 0,
              rate.tp = 1, rate.cc = 0, n1, n2, n3, n4)

cosa.bcrd4f3(score = NULL, dists = "normal", k1 = -6, k2 = 6, rhots = NULL,
             order = 1, interaction = FALSE,
             treat.lower = TRUE, cutoff = 0, p = NULL,
             cn1 = 0, cn2 = 0, cn3 = 0, cn4 = 0, cost = NULL,
             n1 = NULL, n2 = NULL, n3 = NULL, n4 = NULL,
             n0 = c(10, 3, 100 + g3 + order * (1 + interaction), 5), p0 = .499,
             constrain = "power", round = TRUE, max.power = FALSE,
             local.solver = c("LBFGS", "SLSQP"),
             power = .80, es = .25, alpha = .05, two.tailed = TRUE,
             rho2, rho3, g3 = 0, r21 = 0, r22 = 0, r23 = 0)
```

Arguments

`score` vector or list; an empirical score variable or an object with class 'score' returned from the `inspect.score()` function.

`dists` character; distribution of the score variable, `"normal"` or `"uniform"`. By default, `dists = "normal"` specification implies a truncated normal distribution with `k1 = -6` and `k2 = 6`.

`k1` left truncation point for (uncentered) empirical, truncated normal, or uniform distribution. Ignored when `rhots = 0` or `order = 0`.

`k2` right truncation point for (uncentered) empirical, truncated normal, or uniform distribution. Ignored when `rhots = 0` or `order = 0`.

`order` integer >= 0; order of polynomial functional form specification for the score variable.

`interaction` logical; if `TRUE` polynomial specification interacts with the treatment variable.

`rhots` obsolote; use `order = 0` to obtain results equivalent to random assignment designs.

`treat.lower` logical; if `TRUE` units below the cutoff are treated.

`cutoff` decision threshold.

`p` proportion of level 3 units in the treatment condition.

`power` statistical power (1 - β).

`es` effect size (Cohen's d).

`alpha` probability of type I error (α).

`two.tailed` logical; `TRUE` for two-tailed hypothesis testing.

`df` degrees of freedom.

`rho2` proportion of variance in the outcome between level 2 units (unconditional ICC2).

`rho3` proportion of variance in the outcome between level 3 units (unconditional ICC3).

`g3` number of covariates at level 3.

`r21` proportion of level 1 variance in the outcome explained by level 1 covariates.

`r22` proportion of level 2 variance in the outcome explained by level 2 covariates.

`r23` proportion of level 3 variance in the outcome explained by level 3 covariates.

`rate.tp` treatment group participation rate.

`rate.cc` control group crossover rate.

`n1` average number of level 1 units per level 2 unit.

`n2` average number of level 2 units per level 3 unit.

`n3` number of level 3 units(per stratum or block, if exists).

`n4` number of stratum or fixed blocks.

`cn1` marginal costs per level 1 unit in treatment and control conditions (positional), e.g. `c(10, 5)`.

`cn2` marginal costs per level 2 unit in treatment and control conditions (positional), e.g. `c(50, 30)`.

`cn3` marginal costs per level 3 unit in treatment and control conditions (positional), e.g. `c(80, 50)`.

`cn4` marginal cost per stratum or fixed block.

`cost` total cost or budget. Ignored when `constrain = "power"` or `constrain = "es"`.

`p0` starting value for `p` when `rhots = 0` and `p = NULL`. Starting value is replaced with the average when `p` is constrained by bounds.

`n0` vector of starting values for `n1`, `n2`, `n3` or `n1`, `n2`, `n3`, `n4` (positional). Starting values are replaced with the averages when sample sizes are constrained by bounds.

`constrain` character; constrains one of the `"cost"`, `"power"`, or `"es"` at the specified value.

`round` logical; `TRUE` for rounded BCOSSA solution.

`max.power` logical; `TRUE` for maximizing the power rate instead of minimizing the variance. Applies when `constrain = "cost"`.

`local.solver` subset of `c("LBFGS", "SLSQP")`

Value

`parms` list of parameters used in the function.

`df` degrees of freedom.

`sse` standardized standard error.

`cosa` BCOSSA solution.

`mdes` minimum detectable effect size and (1 - α)% confidence limits.

`power` statistical power (1 - β)

Examples

```
score.obj <- inspect.score(rnorm(1000),
                           order = 1, interaction = FALSE,
                           cutoff = 0, k1 = -1, k2 = 1)
# single site (no blocks)
power.crd3(score.obj,
           es = .25, rho2 = .20, rho3 = .10,
           g3 = 0, r23 = 0, n1 = 20, n2 = 3, n3 = 40)

# with 5 blocks (note that r23 is modified but g3 remains the same)
power.bcrd4f3(score.obj,
              es = .25, rho2 = .20, rho3 = .10,
              g3 = 0, r23 = .30,
              n1 = 20, n2 = 3, n3 = 40, n4 = 5)

# minimum required number of level 3 units for each block
cosa.bcrd4f3(score.obj,
             es = .25, rho2 = .20, rho3 = .10,
             g3 = 0, r23 = .30,
             n1 = 20, n2 = 2, n3 = NULL, n4 = 5)
```

`crd4` *Cluster-level Regression Discontinuity (Four-level Design, Discontinuity at Level 4)*

Description

Use mdes.crd4() to calculate minimum detectable effect size, power.crd4() to calculate statistical power, and cosa.crd4() for bound constrained optimal sample size allocation (BCOSSA).

Usage

```
mdes.crd4(score = NULL, dists = "normal", k1 = -6, k2 = 6,
          order = 1, interaction = FALSE,
          treat.lower = TRUE, cutoff = 0, p = NULL,
          power = .80, alpha = .05, two.tailed = TRUE,
          df = n4 - g4 - order * (1 + interaction) - 2,
          rho2, rho3, rho4, r21 = 0, r22 = 0, r23 = 0, r24 = 0,
          g4 = 0, rate.tp = 1, rate.cc = 0, n1, n2, n3, n4)

power.crd4(score = NULL, dists = "normal", k1 = -6, k2 = 6,
           order = 1, interaction = FALSE,
           treat.lower = TRUE, cutoff = 0, p = NULL,
           es = .25, alpha = .05, two.tailed = TRUE,
           df = n4 - g4 - order * (1 + interaction) - 2,
           rho2, rho3, rho4, r21 = 0, r22 = 0, r23 = 0, r24 = 0,
           g4 = 0, rate.tp = 1, rate.cc = 0, n1, n2, n3, n4)

cosa.crd4(score = NULL, dists = "normal", k1 = -6, k2 = 6, rhots = NULL,
          order = 1, interaction = FALSE,
          treat.lower = TRUE, cutoff = 0, p = NULL,
          cn1 = 0, cn2 = 0, cn3 = 0, cn4 = 0, cost = NULL,
          n1 = NULL, n2 = NULL, n3 = NULL, n4 = NULL,
          n0 = c(10, 3, 100, 5 + g4 + order * (1 + interaction)), p0 = .499,
          constrain = "power", round = TRUE, max.power = FALSE,
          local.solver = c("LBFGS", "SLSQP"),
          power = .80, es = .25, alpha = .05, two.tailed = TRUE,
          rho2, rho3, rho4, g4 = 0, r21 = 0, r22 = 0, r23 = 0, r24 = 0)
```

Arguments

score
: vector or list; an empirical score variable or an object with class 'score' returned from the inspect.score() function.

dists
: character; distribution of the score variable, "normal" or "uniform". By default, dists = "normal" specification implies a truncated normal distribution with k1 = -6 and k2 = 6.

k1
: left truncation point for (uncentered) empirical, truncated normal, or uniform distribution. Ignored when rhots = 0 or order = 0.

k2
: right truncation point for (uncentered) empirical, truncated normal, or uniform distribution. Ignored when rhots = 0 or order = 0.

order
: integer >= 0; order of polynomial functional form specification for the score variable.

interaction
: logical; if TRUE polynomial specification interacts with the treatment variable.

`rhots` obsolote; use `order = 0` to obtain results equivalent to random assignment designs.

`treat.lower` logical; if `TRUE` units below the cutoff are treated.

`cutoff` decision threshold.

`p` proportion of level 4 units in the treatment condition.

`power` statistical power (1 - β).

`es` effect size (Cohen's d).

`alpha` probability of type I error (α).

`two.tailed` logical; `TRUE` for two-tailed hypothesis testing.

`df` degrees of freedom.

`rho2` proportion of variance in the outcome between level 2 units (unconditional ICC2).

`rho3` proportion of variance in the outcome between level 3 units (unconditional ICC3).

`rho4` proportion of variance in the outcome between level 4 units (unconditional ICC4).

`g4` number of covariates at level 4.

`r21` proportion of level 1 variance in the outcome explained by level 1 covariates.

`r22` proportion of level 2 variance in the outcome explained by level 2 covariates.

`r23` proportion of level 3 variance in the outcome explained by level 3 covariates.

`r24` proportion of level 4 variance in the outcome explained by level 4 covariates.

`rate.tp` treatment group participation rate.

`rate.cc` control group crossover rate.

`n1` average number of level 1 units per level 2 unit.

`n2` average number of level 2 units per level 3 unit.

`n3` average number of level 3 units per level 4 unit.

`n4` number of level 4 units.

`cn1` marginal costs per level 1 unit in treatment and control conditions (positional), e.g. `c(10, 5)`.

`cn2` marginal costs per level 2 unit in treatment and control conditions (positional), e.g. `c(50, 30)`.

`cn3` marginal costs per level 3 unit in treatment and control conditions (positional), e.g. `c(80, 50)`.

`cn4` marginal costs per level 4 unit in treatment and control conditions (positional), e.g. `c(100, 40)`.

`cost` total cost or budget. Ignored when `constrain = "power"` or `constrain = "es"`.

`p0` starting value for `p` when `rhots = 0` and `p = NULL`. Starting value is replaced with the average when `p` is constrained by bounds.

`n0` vector of starting values for `n1, n2, n3, n4` (positional). Starting values are replaced with the averages when sample sizes are constrained by bounds.

`constrain` character; constrains one of the `"cost"`, `"power"`, or `"es"` at the specified value.

`round` logical; `TRUE` for rounded BCOSSA solution.

`max.power` logical; `TRUE` for maximizing the power rate instead of minimizing the variance. Applies when `constrain = "cost"`.

`local.solver` subset of `c("LBFGS", "SLSQP")`.

Value

parms	list of parameters used in the function.
df	degrees of freedom.
sse	standardized standard error.
cosa	BCOSSA solution.
mdes	minimum detectable effect size and (1 - α)% confidence limits.
power	statistical power (1 - β)

Examples

```
score.obj <- inspect.score(rnorm(1000),
                            order = 1, interaction = FALSE,
                            cutoff = 0, k1 = -1, k2 = 1)

power.crd4(score.obj,
           es = .25, rho2 = .20, rho3 = .10, rho4 = .05,
           g4 = 0, r24 = 0, n1 = 20, n2 = 3, n3 = 50, n4 = 20)

# minimum required number of level 4 units
cosa.crd4(score.obj,
          es = .25, rho2 = .20, rho3 = .10, rho4 = .05,
          g4 = 0, r24 = 0,
          n1 = 20, n2 = 3, n3 = 50, n4 = NULL)
```

inspect.score *Computes Regression Discontinuity Design Effects*

Description

Computes Regression Discontinuity Design Effects (RDDE) either based on analytic deviations (up to second order with interactions), an empirical score variable, or simulation.

Usage

```
  inspect.score(score = NULL, p = NULL, cutoff = NULL,
                treat.lower = FALSE, order = 1, interaction = FALSE,
                mu = 0, sigma = 1, k1 = -Inf, k2 =  Inf,
                dists = "normal", sim = FALSE, ndraw = 1000, nsim = 1000)
```

Arguments

sim	logical; if TRUE results are based on simulation.
score	vector; score variable.
p	proportion of units in the treatment condition.
cutoff	decision threshold.

`treat.lower` logical; if `TRUE` units below cutoff are treated.

`order` integer >= 0; order of polynomial functional form specification for the score variable.

`interaction` logical; if `TRUE` polynomial specification interacts with the treatment variable.

`mu` mean of (uncentered) truncated normal - applies when `score` = `NULL` and `dists` = `"normal"`.

`sigma` standard deviation of (uncentered) truncated normal - applies when `score` = `NULL` and `dists` = `"normal"`.

`k1` left truncation point for (uncentered) empirical, truncated normal, or uniform distribution.

`k2` right truncation point for (uncentered) empirical, truncated normal, or uniform distribution.

`dists` char; type of distribution, `"normal"` or `"uniform"`.

`ndraw` number of draws - applies when `sim` = `TRUE`.

`nsim` number of simulations - applies when `sim` = `TRUE`.

Value

`parms` list; list of parameters used in the computation.

`cutoff` decision threshold (computed if `p` is provided).

`treat.lower` if `TRUE` units below cutoff are treated.

`p` proportion of subjects treated (computed if `cutoff` is provided).

`order` order of polynomial specification for the score variable.

`interaction` if `TRUE` polynomial specification interacts with the treatment variable.

`center` if `TRUE` the score variable is centered on the cutoff

`rdde` regression discontinuity design effect.

Examples

```
# based on an empirical score variable
inspect.score(score = rnorm(10000), cutoff = 0)
# based on analytic derivation
inspect.score(cutoff = 0)
# based on simulation
inspect.score(sim = TRUE, cutoff = 0)
```

ird *Simple Individual-level Regression Discontinuity (w/ or w/o Strata or Fixed Blocks)*

Description

Use mdes.ird() to calculate minimum detectable effect size and power.ird() to calculate statistical power. If higher level strata or fixed blocks exist, use mdes.bird2f1() to calculate minimum detectable effect size, power.bird2f1() to calculate statistical power, and cosa.bird2f1() for bound constrained optimal sample size allocation (BCOSSA).

Usage

```
mdes.ird(score = NULL, dists = "normal", k1 = -6, k2 = 6,
         order = 1, interaction = FALSE,
         treat.lower = TRUE, cutoff = 0, p = NULL,
         power = .80, alpha = .05, two.tailed = TRUE,
         df = n1 - g1 - order * (1 + interaction) - 2,
         r21 = 0, g1 = 0, rate.tp = 1, rate.cc = 0, n1)

power.ird(score = NULL, dists = "normal", k1 = -6, k2 = 6,
          order = 1, interaction = FALSE,
          treat.lower = TRUE, cutoff = 0, p = NULL,
          es = .25, alpha = .05, two.tailed = TRUE,
          df = n1 - g1 - order * (1 + interaction) - 2,
          r21 = 0, g1 = 0, rate.tp = 1, rate.cc = 0, n1)

mdes.bird2f1(score = NULL, dists = "normal", k1 = -6, k2 = 6,
             order = 1, interaction = FALSE,
             treat.lower = TRUE, cutoff = 0, p = NULL,
             power = .80, alpha = .05, two.tailed = TRUE,
             df = n2 * (n1 - 2) - g1 - order * (1 + interaction),
             r21 = 0, g1 = 0, rate.tp = 1, rate.cc = 0, n1, n2 = 1)

power.bird2f1(score = NULL, dists = "normal", k1 = -6, k2 = 6,
              order = 1, interaction = FALSE,
              treat.lower = TRUE, cutoff = 0, p = NULL,
              es = .25, alpha = .05, two.tailed = TRUE,
              df = n2 * (n1 - 2) - g1 - order * (1 + interaction),
              r21 = 0, g1 = 0, rate.tp = 1, rate.cc = 0, n1, n2 = 1)

cosa.bird2f1(score = NULL, dists = "normal", k1 = -6, k2 = 6, rhots = NULL,
             order = 1, interaction = FALSE,
             treat.lower = TRUE, cutoff = 0, p = NULL,
             cn1 = 0, cn2 = 0, cost = NULL,
             n1 = NULL, n2 = NULL,
             n0 = c(400, 5), p0 = .499,
             constrain = "power", round = TRUE, max.power = FALSE,
             local.solver = c("LBFGS", "SLSQP"),
             power = .80, es = .25, alpha = .05, two.tailed = TRUE,
             g1 = 0, r21 = 0)
```

Arguments

score — vector or list; an empirical score variable or an object with class 'score' returned from the inspect.score() function.

dists — character; distribution of the score variable, "normal" or "uniform". By default, dists = "normal" specification implies a truncated normal distribution with k1 = -6 and k2 = 6.

k1 — left truncation point for (uncentered) empirical, truncated normal, or uniform distribution. Ignored when rhots = 0 or order = 0.

k2 — right truncation point for (uncentered) empirical, truncated normal, or uniform distribution. Ignored when rhots = 0 or order = 0.

order — integer >= 0; order of polynomial functional form specification for the score variable.

interaction — logical; if TRUE polynomial specification interacts with the treatment variable.

rhots — obsolote; use order = 0 to obtain results equivalent to random assignment designs.

treat.lower — logical; if TRUE units below cutoff the are treated.

cutoff — decision threshold.

p — proportion of units in the treatment condition.

power — statistical power (1 - β).

es — numeric > 0; effect size (Cohen's d).

alpha — probability of type I error (α).

two.tailed — logical; TRUE for two-tailed hypothesis testing.

df — degrees of freedom.

g1 — number of covariates.

r21 — proportion of variance in the outcome explained by covariates.

rate.tp — treatment group participation rate.

rate.cc — control group crossover rate.

n1 — sample size (per stratum or block, if exists).

n2 — number of stratum or fixed blocks.

cn1 — marginal cost per unit in treatment and control conditions, e.g. c(10, 5).

cn2 — marginal cost per stratum or fixed block.

cost — total cost or budget. Ignored when constrain = "power" or constrain = "es".

constrain — character; constrains one of the "cost", "power", or "es" at the specified value.

n0 — starting value for n1 or n1, n2. Starting value is replaced with the average when sample size is constrained by bounds.

`p0` starting value for `p` when `rhots = 0` and `p = NULL`. Starting value is replaced with average when `p` is constrained by bounds.

`round` logical; `TRUE` for rounded BCOSSA solution.

`max.power` logical; `TRUE` for maximizing power instead of minimizing variance, applies when `constrain = "cost"`

`local.solver` subset of `c("LBFGS", "SLSQP")`

Value

`parms` list of parameters used in the function.

`df` degrees of freedom.

`sse` standardized standard error.

`cosa` BCOSSA solution.

`mdes` minimum detectable effect size and (1 - α)% confidence limits.

`power` statistical power (1 - β)

Examples

```
score.obj <- inspect.score(rnorm(1000),
                           order = 1, interaction = FALSE,
                           cutoff = 0, k1 = -1, k2 = 1)

# single site (no blocks)
power.ird(score.obj, g1 = 0, r21 = 0,
          es = 0.25, n = 100)

# with 5 blocks (note that r21 is modified but g1 remains the same)
power.bird2f1(score.obj,  g1 = 0, r21 = .30,
              es = 0.25, n1 = 100, n2 = 5)

# minimum required sample size for each block
cosa.bird2f1(score.obj,  g1 = 0, r21 = .30,
             n1 = NULL, n2 = 5)
```

`moments` *Moments*

Description

If data (vector) is provided use `emp.moment()` function, otherwise for truncated normal distribution use `tnorm.moment()`, and for uniform distribution use `unif.moment()`.

Usage

```
tnorm.moment(mu = 0, sigma = 1, k1 = -Inf, k2 = Inf, order = 1, central = FALSE)
unif.moment(k1 = 0, k2 = 1, order = 1, central = FALSE)
emp.moment(x, order = 1, central = FALSE, absolute = FALSE, na.rm = FALSE)
```

Arguments

`mu` mean of truncated normal - applies to `tnorm.moment()`.

`sigma` standard deviation of truncated normal - applies to `tnorm.moment()`.

`k1` left truncation point for truncated normal distribution or lower bound for uniform distribution.

`k2` right truncation point for truncated normal distribution or upper bound for uniform distribution.

`order` + int; order of moment

`x` a vector of values - applies to `emp.moment()`.

`central` logical; if `TRUE` produces central moments.

`absolute` logical; if `TRUE` produces absolute moments - applies to `emp.moment()`.

`na.rm` logical; if `TRUE` removes missing values - applies to `emp.moment()`.

Examples

```
tnorm.moment(k1 = -20, k2 = 20, order = 4, central = FALSE)
emp.moment(rnorm(10000), order = 4, central = FALSE)
unif.moment(k1 = 0, k2 = 1, order = 4, central = FALSE)
emp.moment(runif(10000), order = 4, central = FALSE)
```

`plot` *Power and MDES Curves*

Description

Plots statistical power or minimum detectable effect size curves with (1-α)x100 % confidence interval for the design of interest.

Usage

```
## S3 method for class 'power'
plot(x, score = NULL, ypar = "mdes",  xpar = NULL,
          xlim = NULL, ylim = NULL,
          xlab = NULL, ylab = NULL,
          main = NULL, sub = NULL,
          locate = FALSE, benchmark = NULL, ...)

## S3 method for class 'mdes'
plot(x, score = NULL, ypar = "mdes",  xpar = NULL,
          xlim = NULL, ylim = NULL,
          xlab = NULL, ylab = NULL,
          main = NULL, sub = NULL,
          locate = FALSE, benchmark = NULL, ...)
```

```
## S3 method for class 'cosa'
plot(x, score = NULL, ypar = "mdes",  xpar = NULL,
          xlim = NULL, ylim = NULL,
          xlab = NULL, ylab = NULL,
          main = NULL, sub = NULL,
          locate = FALSE, benchmark = NULL, ...)
```

Arguments

`x` an object returned from functions in **cosa** package.

`score` vector or list; an empirical score variable or an object with class ’score’ returned from the `inspect.score()` function.

`ypar` character; `"mdes"` or `"power"` on `y` axis.

`xpar` character; one of the sample sizes on `x` axis.

`xlim` limits for `xpar`.

`ylim` limits for `ypar`.

`xlab` `x` axis label.

`ylab` `y` axis label.

`main` title for the plot.

`sub` subtitle for the plot.

`locate` logical; `TRUE` locates parameter values for design `x` on the plot.

`benchmark` benchmark line.

`...` other graphical parameters to pass to `plot.new()`.

Examples

```
d1 <- mdes.bcrd3r2(rho2 = .10, rho3 = .20, omega3 = .30,
                   n1 = 20, n2 = 44, n3 = 50)
plot(d1, xpar = "n3", xlim = c(30, 100))
```

Vectorize BCOSSA *Vectorizes BCOSSA Solutions*

Description

Vectorizes bound constrained optimal sample size allocation (BCOSSA) solutions based on multiple sets of parameter values. This is particularly useful when multiple values of design parameters are to be considered.

Usage

```
vectorize.cosa(x, score = NULL,
             args.grid, args.names = NULL,
             ordered = TRUE, ncase = 10L)
```

Arguments

`x` an object returned from one of the `cosa.<design>()` functions.

`score` vector or list; an empirical score variable or an object with class 'score' returned from the `inspect.score()` function.

`args.grid` vector or matrix: arguments' grid consisting of sets of parameter values. A vector of values (for a single parameter) or a matrix (for multiple parameters).

`args.names` character list; arguments' names. Default option `args.names = NULL` uses column names from `args.grid`

`ordered` logical: whether results should be ordered (cases with worst power rate or highest total cost are on top).

`ncase` integer: number of cases to be subsetted, ignored if `ordered = FALSE`.

Examples

```
design <- cosa.crd2(order = 0, round = FALSE,
                    constrain = "power", power = .80,
                    cn1 = c(20, 10), cn2 = c(200, 50),
                    es = .25, rho2 = .10,
                    g2 = 3, r22 = .30,
                    n1 = NULL, n2 = NULL, p = NULL)

args.grid <- expand.grid(
  rho2 = seq(.15, .25, .05)
)

vectorize.cosa(design, args.grid = args.grid, ordered = FALSE)
```

Index