Package 'MedianaDesigner'

June 24, 2022

Type Package

Title Efficient Simulation-Based Power and Sample Size Calculations for a Broad Class of Late-Stage Clinical Trials

Version 0.8

Date 2022-06-20

Author Alex Dmitrienko [aut, cre]

Depends R (>= 3.1.2)

Imports Rcpp (>= 0.12.10), RcppNumerical, methods, officer, flextable, devEMF, mvtnorm, shiny, shinydashboard, shinyMatrix, foreach, parallel, doParallel, doRNG, MASS, rootSolve, lme4, lmerTest, pbkrtest

Suggests testthat

LinkingTo Rcpp, RcppEigen, RcppNumerical

Maintainer Alex Dmitrienko <admitrienko@mediana.us>

Description The following modules are included in the package:
Adaptive designs with data-driven sample size or event count re-estimation,
Adaptive designs with data-driven treatment selection,
Adaptive designs with data-driven population selection,
Optimal selection of a futility stopping rule,
Event prediction in event-driven trials,
Adaptive trials with response-adaptive randomization (experimental module),
Traditional trials with multiple objectives (experimental module).
Traditional trials with cluster-randomized designs (experimental module).

License GPL-3

LazyLoad yes

LazyData true

NeedsCompilation yes

Repository CRAN

Date/Publication 2022-06-24 16:00:08 UTC

URL https://github.com/medianasoft/MedianaDesigner

BugReports https://github.com/medianasoft/MedianaDesigner/issues

R topics documented:

MedianaDesigner-package

Efficient Simulation-Based Power and Sample Size Calculations for a Broad Class of Late-Stage Clinical Trials

Description

The MedianaDesigner package provides efficient simulation-based power and sample size calculations for a broad class of late-stage clinical trials, including Phase II trials, seamless Phase II/III trials and Phase III trials:

- Adaptive trials with data-driven sample size or event count re-estimation.
- Adaptive trials with data-driven treatment selection.
- Adaptive trials with data-driven population selection.
- Optimal selection of a futility stopping rule.
- Blinded event prediction in event-driven trials.
- Adaptive trials with response-adaptive randomization (experimental module).
- Traditional trials with multiple objectives (experimental module).
- Traditional trials with cluster-randomized designs (experimental module).

For more information on this package, visit https://mediana.us/free-software/. The technical manuals with a detailed description of the statistical methodology implemented in each module are available at https://medianasoft.github.io/MedianaDesigner. Additional information and multiple case studies can be found in the online manual at https://medianasoft.github.io/MedianaDesigner.

Details

Package:	MedianaDesigner
Type:	Package
Version:	0.8
Date:	2022-06-20
License:	GPL-3

Key functions included in the package:

- ADSSMod: Simulation-based design of adaptive trials with data-driven sample size or event count re-estimation.
- ADSSModApp: Graphical user interface for designing adaptive trials with data-driven sample size or event count re-estimation.
- ADTreatSel: Simulation-based design of adaptive trials with data-driven treatment selection.
- ADTreatSelApp: Graphical user interface for designing adaptive trials with data-driven treatment selection.

- ADPopSel: Simulation-based design of adaptive trials with data-driven population selection.
- ADPopSelApp: Graphical user interface for designing adaptive trials with data-driven population selection.
- FutRule: Simulation-based selection of an optimal futility stopping rule at an interim analysis.
- FutRuleApp: Graphical user interface for an optimal selection of a futility stopping rule.
- EventPred: Blinded simulation-based event prediction in trials with an event-driven design.
- EventPredPriorDistribution: Calculation of the parameters of prior gamma distributions.
- EventPredApp: Graphical user interface for event prediction.
- ADRand: Simulation-based design of adaptive trials with response-adaptive randomization.
- ADRandApp: Graphical user interface for designing adaptive trials with response-adaptive randomization.
- MultAdj: Simulation-based design of traditional trials with multiple objectives.
- MultAdjApp: Graphical user interface for power calculations in traditional trials with multiple objectives.
- ClustRand: Simulation-based design of cluster-randomized trials.
- ClustRandApp: Graphical user interface for power calculations in cluster-randomized trials.
- GenerateReport: Simulation report for any module.

The package comes with one example data set:

- EventPredData: Example data set to illustrate event prediction in event-driven trials.

Multiple case studies are included in this vignette to illustrate the use of this package for designing clinical trials with data-driven design elements:

- ADSSModExample1: Simulation-based design of an adaptive trial with sample size re-estimation (normally distributed endpoint).
- ADSSModExample2: Simulation-based design of an adaptive trial with sample size re-estimation (binary endpoint).
- ADSSModExample3: Simulation-based design of an adaptive trial with event count re-estimation (time-to-event endpoint).
- ADTreatSelExample1: Simulation-based design of an adaptive trial with treatment selection (normally distributed endpoint).
- ADTreatSelExample2: Simulation-based design of an adaptive trials with treatment selection (binary endpoint).
- ADTreatSelExample3: Simulation-based design of an adaptive trials with treatment selection (time-to-event endpoint).
- ADPopSelExample1: Simulation-based design of an adaptive trial with population selection (normally distributed endpoint).
- ADPopSelExample2: Simulation-based design of an adaptive trials with population selection (binary endpoint).
- ADPopSelExample3: Simulation-based design of an adaptive trials with population selection (time-to-event endpoint).

- FutRuleExample1: Simulation-based selection of an optimal futility stopping rule (normally distributed endpoint).
- FutRuleExample2: Simulation-based selection of an optimal futility stopping rule (binary endpoint).
- FutRuleExample3: Simulation-based selection of an optimal futility stopping rule (time-to-event endpoint).
- EventPredExample: Simulation-based event prediction in trials with an event-driven design (time-to-event endpoint).
- ADRandExample: Simulation-based design of an adaptive dose-finding trial with response-adaptive randomization (normally distributed endpoint).
- MultAdjExample1: Simulation-based power calculations in Phase III trials with multiple dose-placebo comparisons.
- MultAdjExample2: Simulation-based power calculations in Phase III trials with multiple endpoints.
- MultAdjExample3: Simulation-based power calculations in Phase III trials with multiple endpoints and multiple dose-placebo comparisons.
- ClustRandExample1: Simulation-based design of a cluster-randomized trial (normally distributed endpoint).
- ClustRandExample2: Simulation-based design of a cluster-randomized trial (binary endpoint).

Author(s)

Mediana (www.mediana.us). We would like to thank multiple individuals, including Thomas Brechenmacher (IQVIA), Douglas McNair (Gates Foundation) and Thomas Peppard (Certara), for their feedback that helped us improve the package and add new features.

References

Ahn, C., Heo, M., Zhang, S. (2015). *Sample Size Calculations for Clustered and Longitudinal Outcomes in Clinical Research*. Chapman and Hall/CRC.

Alosh, M., Bretz, F., Huque, M. (2014). Advanced multiplicity adjustment methods in clinical trials. *Statistics in Medicine*. 33, 693-713.

Bagiella, E., Heitjan, D.F. (2001). Predicting analysis times in randomized clinical trials. *Statistics in Medicine*. 20, 2055-2063.

Chuang-Stein, C., Kirby, S., French, J., Kowalski, K., Marshall, S., Smith, M. K. (2011). A quantitative approach for making go/no-go decisions in drug development. *Drug Information Journal*. 45, 187-202.

Dmitrienko, A., Bretz, F., Westfall, P.H., et al. (2009). Multiple testing methodology. *Multiple testing problems in pharmaceutical statistics*. Dmitrienko, A., Tamhane, A.C., Bretz, F. (editors). New York: Chapman and Hall/CRC Press.

Dmitrienko, A., Tamhane, A.C. (2011). Mixtures of multiple testing procedures for gatekeeping applications in clinical trials. *Statistics in Medicine*. 30, 1473-1488.

Dmitrienko, A., D'Agostino, R. Sr. (2013). Traditional multiplicity adjustment methods in clinical trials. *Statistics in Medicine*. 32, 5172-5218.

Dmitrienko, A., Kordzakhia, G., Brechenmacher, T. (2016). Mixture-based gatekeeping procedures for multiplicity problems with multiple sequences of hypotheses. *Journal of Biopharmaceutical Statistics*. 26, 758-780.

Dmitrienko, A., Paux, G. (2017). Subgroup analysis in clinical trials. *Clinical Trial Optimization Using R*. Dmitrienko, A., Pulkstenis, E. (editors). Chapman and Hall/CRC Press, New York.

Dmitrienko, A., D'Agostino, R.B. (2018). Multiplicity considerations in clinical trials. *New England Journal of Medicine*. 378, 2115-2122.

Hayes, R.J., Moulton, L.H. (2009). emphCluster Randomised Trials: A Practical Approach. Chapman and Hall/CRC.

Herson, J. (1979). Predictive probability early termination plans for Phase II clinical trials. *Biometrics*. 35, 775-783.

Kordzakhia, G., Brechenmacher, T., Ishida, E., Dmitrienko, A., Zheng, W.W., Lie, D.F. (2018). An enhanced mixture method for constructing gatekeeping procedures in clinical trials. *Journal of Biopharmaceutical Statistics*. 28, 113-128.

Millen, B., Dmitrienko, A., Ruberg, S., Shen, L. (2012). A statistical framework for decision making in confirmatory multi-population tailoring clinical trials. *Drug Information Journal*. 46, 647-656.

Wang, D., Cui, L., Zhang, L., Yang, B. (2014). An ROC approach to evaluate interim go/no-go decision-making quality with application to futility stopping in the clinical trial designs. *New Developments in Statistical Modeling, Inference and Application*. Jin, Z., Liu, M., Luo, X. (editors). Springer, New York. 121-147.

Wassmer, G., Brannath, W. (2016). *Group Sequential and Confirmatory Adaptive Designs in Clinical Trials*. New York: Springer.

`ADPopSel` *Simulation-based design of adaptive trials with data-driven population selection*

Description

This function performs a simulation-based evaluation of operating characteristics for adaptive trials with data-driven population selection. A two-arm Phase III clinical trial with two pre-defined patient populations (overall population and subpopulation of biomarker-positive patients) and two interim analyses is assumed. The first interim analysis supports early stopping for futility in the overall population and the second interim analysis identifies the most promising patient population or populations. For examples of the function call, see ADPopSelExample1, ADPopSelExample2 or ADPopSelExample3.

Usage

```
ADPopSel(parameters)
```

Arguments

`parameters` List of the trial design and other parameters. The required elements are defined below:

- `endpoint_type`: Character value defining the primary endpoint's type. Possible values:
  - `"Normal"`: Normally distributed endpoint.
  - `"Binary"`: Binary endpoint.
  - `"Time-to-event"`: Time-to-event endpoint.
- `direction`: Character value defining the direction of favorable outcome. Possible values: `"Higher"` (a higher value of the endpoint indicates a more favorable outcome) and `"Lower"` (a lower value of the endpoint indicates a more favorable outcome).
- `sample_size`: Integer vector with two elements defining the number of enrolled patients in the two trial arms (control and experimental treatment). Each element must be positive.
- `control_mean`: Numeric value defining the mean of the primary endpoint in the control arm. This parameter is required only with normally distributed endpoints (`endpoint_type="Normal"`).
- `control_sd`: Numeric value defining the standard deviation of the primary endpoint in the control arm. This value must be positive. This parameter is required only with normally distributed endpoints.
- `treatment_mean`: Numeric value defining the mean of the primary endpoint in the experimental treatment arm. This parameter is required only with normally distributed endpoints.
- `treatment_sd`: Numeric value defining the standard deviation of the primary endpoint in the experimental treatment arm. This value must be positive. This parameter is required only with normally distributed endpoints.
- `control_rate`: Numeric value defining the proportion or response rate for the primary endpoint in the control arm. This value must be between 0 and 1. This parameter is required only with binary endpoints (`endpoint_type="Binary"`).
- `treatment_rate`: Numeric value defining the proportion or response rate for the primary endpoint in the experimental treatment arm. This value must be between 0 and 1. This parameter is required only with binary endpoints.
- `control_time`: Numeric value defining the median time, e.g., the median survival time, for the exponentially distributed primary endpoint in the control arm. This value must be positive. This parameter is required only with time-to-event endpoints (`endpoint_type="Time-to-event"`).
- `treatment_time`: Numeric value defining the median time, e.g., the median survival time, for the exponentially distributed primary endpoint in the experimental treatment arm. This value must be positive. This parameter is required only with time-to-event endpoints.
- `info_frac`: Numeric vector with three elements defining the information fractions at the first interim analysis, at the second interim analysis and at the final analysis. The first and second elements must be between 0 and 1, and the third element must be 1.

- `event_count`: Numeric value defining the total number of events at the final analysis. This value must be positive. This parameter is required only with time-to-event endpoints.
- `futility_threshold`: Numeric value defining the futility threshold for conditional power at the first interim analysis. This value must be positive.
- `influence`: Numeric value defining the influence threshold for selecting the most promising population or populations at the second interim analysis. This value must be positive.
- `interaction`: Numeric value defining the interaction threshold for selecting the most promising population or populations at the second interim analysis. This value must be greater than 1.
- `dropout_rate`: Numeric value defining the patient dropout rate. With normally distributed endpoints and binary endpoints, a uniform patient dropout process is assumed and thus this parameter defines the fraction of patients that will be excluded from the interim and final analyses. With time-to-event endpoints, the time to patient dropout is assumed to follow an exponential distribution and this parameter defines the annual dropout rate. This value must be between 0 and 1.
- `enrollment_period`: Numeric value defining the length of the patient enrollment period. This value must be positive. This parameter is required only with time-to-event endpoints.
- `enrollment_parameter`: Numeric value defining the median enrollment time. The patient enrollment process is assumed to be governed by a truncated exponential distribution and this parameter defines the time point by which 50% of the patients are enrolled into the trial. This value must be between 0 and the length of the patient enrollment period. This parameter is required only with time-to-event endpoints..
- `alpha`: Numeric value defining the overall one-sided Type I error rate. The default value is 0.025.
- `random_seed`: Integer value defining the random number generator seed. The default value is 49283.
- `nsims`: Integer value defining the number of simulation runs.
- `ncores`: Integer value defining the number of cores for parallel calculations. The number of cores cannot exceed the maximum available number of cores. The default value is 1.

Value

The function returns an object of class '`ADPopSelResults`'. This object is a list with the following components:

`parameters`	List containing the user-specified parameters.
`sim_results`	Data frame containing key descriptive statistics for each simulation run.
`sim_summary`	List containing the key operating characteristics of the adaptive design.

A detailed summary of the simulation results can be created using the `GenerateReport` function.

See Also

ADPopSelApp

ADPopSelApp *Graphical user interface to design an adaptive trial with data-driven population selection*

Description

This function creates a web application with a Shiny-based graphical user interface to facilitate the process of designing Phase III adaptive trials with two interim analysis to enable a futility assessment and data-driven selection of the most promising patient population or populations. The application supports the same functionality as the ADPopSel function. For an example of the function call, see ADPopSelExample1.

Usage

```
ADPopSelApp()
```

Arguments

No arguments

Value

No return value, called for side effects

See Also

ADPopSel

ADPopSelExample1 *Simulation-based design of an adaptive trial with population selection (normally distributed endpoint)*

Description

Consider a seamless Phase II/Phase III or Phase III trial with a normally distributed primary efficacy endpoint (a larger value of this endpoint indicates a more favorable outcome). The efficacy and safety profiles of a single dose or regimen of an experimental treatment will be evaluated versus a control in two patient populations, namely, the overall population of patients and a pre-defined subset of patients with a biomarker-positive status. An adaptive design with two interim analyses will be used in the trial (the first interim analysis may correspond to the end of the Phase II portion of the trial) to support the following decision rules:

- Futility stopping rule will be applied at the first interim analysis (IA1). The trial will be terminated for futility if the predicted probability of success (conditional power) in the overall population is too low
- Population selection rule will be applied at the second interim analysis (IA2). The most promising population or populations will be selected for the final analysis (FA).

The following design parameters will be assumed:

- A balanced design with 220 enrolled patients per arm will be utilized in this trial.
- The prevalence of biomarker-positive patients in the overall population is expected to be 50%.
- The patient dropout rate at the end of the treatment period is equal to 10%.
- The information fractions at IA1 and IA2 are set to 40% and 60%, i.e., the first and second interim looks will be taken after 40% and 60% of the patients complete the treatment period or drop out of the trial before completing the treatment period.

In addition, the following decision rules will be considered at the two interim analyses:

- The futility threshold at IA1 is set to 20%.
- The influence and interaction thresholds used in the population selection rule are set to 0.1 and 1.3. The conditions based on these thresholds are applied sequentially to determine if the final analysis should be performed in the overall population only, in the biomarker-positive population only or simultaneously in both populations.

The treatment effect assumptions need to be specified within the biomarker-positive population as well as the complementary population (biomarker-negative population). A common effect will be assumed in the control arm, i.e., the mean and standard deviation of the primary efficacy endpoint will be 0 and 1, respectively, in the biomarker-positive and biomarker-negative populations. A stronger treatment effect will be assumed in the biomarker-positive subset compared to the complementary subset, i.e., the mean of 0.4 in biomarker-positive patients and the mean 0.25 in biomarker-negative patients with a common standard deviation of 1.

Key operating characteristics of the proposed adaptive design with population selection will be evaluated using the `ADPopSel` function based on 10,000 simulation runs. To invoke this function, a list of all design and decision rule parameters (`parameters`) needs to be created as shown below. A comprehensive simulation report for this adaptive design can be generated by calling the `GenerateReport` function and a graphical user interface can be launched using the `ADPopSelApp` function.

Arguments

No arguments

Value

No return value

See Also

`ADPopSel`, `ADPopSelExample2`, `ADPopSelExample3`

Examples

```
# List of all parameters
parameters = list()

# Endpoint type
parameters$endpoint_type = "Normal"

# Direction of favorable outcome
parameters$direction = "Higher"

# Number of enrolled patients (control, treatment)
parameters$sample_size = c(220, 220)

# Prevalence of biomarker-positive patients in the overall population
parameters$prevalence = 0.5

# Patient dropout rate
parameters$dropout_rate = 0.1

# Mean and SD in the control arm (biomarker-negative, biomarker-positive)
parameters$control_mean = c(0, 0)
parameters$control_sd = c(1, 1)

# Mean and SD in the treatment arm (biomarker-negative, biomarker-positive)
parameters$treatment_mean = c(0.25, 0.4)
parameters$treatment_sd = c(1, 1)

# Information fractions at IA1, IA2, FA
parameters$info_frac = c(0.4, 0.6, 1)

# Futility threshold for conditional power at IA1
parameters$futility_threshold = 0.2

# Influence threshold at IA2
parameters$influence = 0.1

# Interaction threshold at IA2
parameters$interaction = 1.3

# One-sided alpha level
parameters$alpha = 0.025

# Number of simulations, you should prefer more
parameters$nsims = 100

# Run simulations to compute operating characteristics
results = ADPopSel(parameters)

# Generate a simulation report (remove tempfile)
GenerateReport(results,
               tempfile("ADPopSel Normally distributed endpoint.docx", fileext=".docx"))
```

`ADPopSelExample2` *Simulation-based design of an adaptive trial with population selection (binary endpoint)*

Description

Consider a seamless Phase II/Phase III or Phase III trial with a binary primary efficacy endpoint with a higher proportion indicating a more favorable outcome. The efficacy and safety profiles of a single dose or regimen of an experimental treatment will be evaluated versus a control in two patient populations (overall population and biomarker-positive population). An adaptive design with two interim analyses will be used in the trial (the first interim analysis may correspond to the end of the Phase II portion of the trial) to support the following decision rules:

- Futility stopping rule will be applied at the first interim analysis (IA1). The trial will be terminated for futility if the predicted probability of success (conditional power) in the overall population is too low
- Population selection rule will be applied at the second interim analysis (IA2). The most promising population or populations will be selected for the final analysis (FA).

The following design parameters will be assumed:

- A balanced design with 120 enrolled patients per arm will be utilized in this trial.
- The prevalence of biomarker-positive patients in the overall population is expected to be 40%.
- The patient dropout rate at the end of the treatment period is equal to 15%.
- The information fractions at IA1 and IA2 are set to 40% and 60%, i.e., the first and second interim looks will be taken after 40% and 60% of the patients complete the treatment period or drop out of the trial before completing the treatment period.

In addition, the following decision rules will be considered at the two interim analyses:

- The futility threshold at IA1 is set to 20%.
- The influence and interaction thresholds used in the population selection rule are set to 0.1 and 1.3. The conditions based on these thresholds are applied sequentially to determine if the final analysis should be performed in the overall population only, in the biomarker-positive population only or simultaneously in both populations.

The treatment effect assumptions are specified within the biomarker-positive and biomarker-negative populations. A common effect will be assumed in the control arm, i.e., the response rate of 10% regardless of the biomarker status, and a strong differential treatment effect will be considered. Specifically, a fairly weak response rate of 25% will be assumed in patients with a biomarker-negative status and a much stronger effect with a response rate of 40% in patients with a biomarker-positive status.

Key operating characteristics of the proposed adaptive design with population selection will be evaluated using the `ADPopSel` function based on 10,000 simulation runs. To invoke this function, a list of all design and decision rule parameters (`parameters`) needs to be created as shown below. A comprehensive simulation report for this adaptive design can be generated by calling the `GenerateReport` function and a graphical user interface can be launched using the `ADPopSelApp` function.

Arguments

No arguments

Value

No return value

See Also

ADPopSel, ADPopSelExample1, ADPopSelExample3

Examples

```
# List of all parameters
parameters = list()

# Endpoint type
parameters$endpoint_type = "Binary"

# Direction of favorable outcome
parameters$direction = "Higher"

# Number of enrolled patients (control, treatment)
parameters$sample_size = c(120, 120)

# Prevalence of biomarker-positive patients in the overall population
parameters$prevalence = 0.4

# Patient dropout rate
parameters$dropout_rate = 0.15

# Response rate in the control arm (biomarker-negative, biomarker-positive)
parameters$control_rate = c(0.1, 0.1)

# Response rate in the treatment arm (biomarker-negative, biomarker-positive)
parameters$treatment_rate = c(0.25, 0.4)

# Information fractions at IA1, IA2, FA
parameters$info_frac = c(0.4, 0.6, 1)

# Futility threshold for conditional power at IA1
parameters$futility_threshold = 0.2

# Influence threshold at IA2
parameters$influence = 0.1

# Interaction threshold at IA2
parameters$interaction = 1.3

# One-sided alpha level
parameters$alpha = 0.025
```

```
# Number of simulations, you should prefer more
parameters$nsims = 100

# Run simulations to compute key characteristics
results = ADPopSel(parameters)

# Generate a simulation report (remove tempfile)
GenerateReport(results,
               tempfile("ADPopSel Binary endpoint.docx", fileext=".docx"))
```

ADPopSelExample3 *Simulation-based design of an adaptive trial with population selection (time-to-event endpoint)*

Description

Consider a seamless Phase II/Phase III or Phase III trial with a binary primary efficacy endpoint. The efficacy and safety profiles of a single dose or regimen of an experimental treatment will be evaluated versus a control in two patient populations (overall population and biomarker-positive population). An adaptive design with two interim analyses will be used in the trial (the first interim analysis may correspond to the end of the Phase II portion of the trial) to support the following decision rules:

- Futility stopping rule will be applied at the first interim analysis (IA1). The trial will be terminated for futility if the predicted probability of success (conditional power) in the overall population is too low
- Population selection rule will be applied at the second interim analysis (IA2). The most promising population or populations will be selected for the final analysis (FA).

The following design parameters will be assumed:

- A balanced design with 230 enrolled patients per arm will be utilized in this trial.
- The prevalence of biomarker-positive patients in the overall population is expected to be 40%.
- The target number of events in the overall population at FA is set to 300. In addition, the target number of events needs to be defined in the biomarker-positive population in case this population is chosen for the final assessment, this value is equal to 150.
- The annual patient dropout rate is equal to 5% (the time to patient dropout is assumed to follow an exponential distribution).
- The length of the patient enrollment period is 12 months and the median enrollment time is expected to be 8 months.
- The information fractions at IA1 and IA2 are set to 40% and 60%, i.e., the first and second interim looks will be taken after 40% and 60% of the patients complete the treatment period or drop out of the trial before completing the treatment period.

In addition, the following decision rules will be considered at the two interim analyses:

- The futility threshold at IA1 is set to 10%.

- The influence and interaction thresholds used in the population selection rule are set to 0.1 and 1.3. The conditions based on these thresholds are applied sequentially to determine if the final analysis should be performed in the overall population only, in the biomarker-positive population only or simultaneously in both populations.

The treatment effect assumptions are specified within the biomarker-positive and biomarker-negative populations. A common effect will be assumed in the control arm, i.e., the median time of 7.5 months regardless of the biomarker status. The median time of 9.5 months in the treated patients with a biomarker-negative status indicates a weaker treatment effect (hazard ratio of 0.79) whereas the median time of 11 months in the treated patients with a biomarker-positive status corresponds to a much stronger treatment effect (hazard ratio of 0.68).

Key operating characteristics of the proposed adaptive design with population selection will be evaluated using the ADPopSel function based on 10,000 simulation runs. To invoke this function, a list of all design and decision rule parameters (parameters) needs to be created as shown below. A comprehensive simulation report for this adaptive design can be generated by calling the GenerateReport function and a graphical user interface can be launched using the ADPopSelApp function.

Arguments

No arguments

Value

No return value

See Also

ADPopSel, ADPopSelExample1, ADPopSelExample2

Examples

```
# List of all parameters
parameters = list()

# Endpoint type
parameters$endpoint_type = "Time-to-event"

# Direction of favorable outcome
parameters$direction = "Higher"

# Number of enrolled patients (control, treatment)
parameters$sample_size = c(230, 230)

# Prevalence of biomarker-positive patients in the overall population
parameters$prevalence = 0.4

# Annual patient dropout rate
parameters$dropout_rate = 0.05

# Median times in the control arm (biomarker-negative, biomarker-positive)
```

```
parameters$control_time = c(7.5, 7.5)

# Median times in the treatment arm (biomarker-negative, biomarker-positive)
parameters$treatment_time = c(9.5, 11)

# Information fractions at IA1, IA2, FA
parameters$info_frac = c(0.4, 0.6, 1)

# Target event count at FA (overall, biomarker-positive)
parameters$event_count = c(300, 150)

# Futility threshold for conditional power at IA1
parameters$futility_threshold = 0.1

# Influence threshold at IA2
parameters$influence = 0.1

# Interaction threshold at IA2
parameters$interaction = 1.3

# Enrollment period
parameters$enrollment_period = 12

# Median enrollment time
parameters$enrollment_parameter = 8

# One-sided alpha level
parameters$alpha = 0.025

# Number of simulations, you should prefer more
parameters$nsims = 100

# Run simulations to compute operating characteristics
results = ADPopSel(parameters)

# Generate a simulation report (remove tempfile)
GenerateReport(results,
               tempfile("ADPopSel Time-to-event endpoint.docx", fileext=".docx"))
```

ADRand *Simulation-based design of adaptive trials with response-adaptive randomization*

Description

This function performs a simulation-based evaluation of operating characteristics for multi-stage adaptive Phase II trials that employ response-adaptive randomization. Patients are assigned to receive placebo or one of several doses of the experimental treatment. The efficacy data at the end of each stage are used to update the randomization scheme for the next stage with the ultimate goal of assigning most patients to the most effective doses. For an example of the function call, see ADRandExample.

Usage

```
ADRand(parameters)
```

Arguments

`parameters` List of the trial design and other parameters. The required elements are defined below:

- `endpoint_type`: Character value defining the primary endpoint's type. Possible values:
  - `"Normal"`: Normally distributed endpoint.
- `direction`: Character value defining the direction of favorable outcome. Possible values: `"Higher"` (a higher value of the endpoint indicates a more favorable outcome) and `"Lower"` (a lower value of the endpoint indicates a more favorable outcome).
- `dose_levels`: Integer vector defining the dose levels in the trial (0 corresponds to the placebo group). Each element must be non-negative.
- `stage_sample_size`: Integer vector defining the total number of patients enrolled in each stage. Each element must be positive.
- `control_mean`: Numeric value defining the mean of the primary endpoint in the placebo arm.
- `control_sd`: Numeric value defining the standard deviation of the primary endpoint in the placebo arm. This value must be positive.
- `treatment_mean`: Numeric vector defining the mean of the primary endpoint in each dosing arm.
- `treatment_sd`: Numeric vector defining the standard deviation of the primary endpoint in each dosing arm.
- `treatment_period`: Numeric value defining the length of the treatment period in the trial.
- `ratio_placebo`: Numeric value defining the fixed randomization ratio in the placebo arm. This value must be between 0 and 1.
- `balance`: Numeric value defining the degree of balance for adaptive randomization. This value must be between 0 and 3.
- `delta`: Numeric value defining the threshold for a clinically meaningful improvement over placebo.
- `exponential_model_parameter`: Numeric value defining the non-linear parameter (delta) for the exponential dose-response model used in the MCPMod method. This value must be positive.
- `emax_model_parameter`: Numeric value defining the non-linear parameter (ED50) for the Emax dose-response model used in the MCPMod method. This value must be positive.
- `logistic_model_parameters`: Numeric vector with two elements defining the non-linear parameters (ED50 and delta) for the logistic dose-response model used in the MCPMod method. The values must be positive.
- `enrollment_period`: Numeric value defining the length of the patient enrollment period. This value must be positive.

- `enrollment_parameter`: Numeric value defining the median enrollment time. The patient enrollment process is assumed to be governed by a truncated exponential distribution and this parameter defines the time point by which 50% of the patients are enrolled into the trial. This value must be between 0 and the length of the patient enrollment period.
- `dropout_rate`: Numeric value defining the patient dropout rate. A uniform patient dropout process is assumed and thus this parameter defines the fraction of patients that will be excluded from the interim and final analyses. This value must be between 0 and 1.
- alpha: Numeric value defining the overall one-sided Type I error rate. The default value is 0.025.
- `random_seed`: Integer value defining the random number generator seed. The default value is 49283.
- `nsims`: Integer value defining the number of simulation runs.
- `ncores`: Integer value defining the number of cores for parallel calculations. The number of cores cannot exceed the maximum available number of cores. The default value is 1.

Value

The function returns an object of class '`ADRandResults`'. This object is a list with the following components:

`parameters`	List containing the user-specified parameters.
`sim_results`	List with multiple data frames containing key descriptive statistics for each simulation run.

A detailed summary of the simulation results can be created using the `GenerateReport` function.

See Also

`ADRandApp`

`ADRandApp` *Graphical user interface to design an adaptive trial with data-driven population selection*

Description

This function creates a web application with a Shiny-based graphical user interface to facilitate the process of designing dose-finding Phase II trials with response-adaptive randomization. The application supports the same functionality as the `ADRand` function. For an example of the function call, see ADRandExample.

Usage

```
ADRandApp()
```

Arguments

No arguments

Value

No return value, called for side effects

See Also

ADRand

ADRandExample *Simulation-based design of dose-finding Phase II trials with response-adaptive randomization (normally distributed endpoint)*

Description

Consider a Phase II trial to be conducted to evaluate the efficacy and safety profiles of several doses of an experimental treatment compared to placebo. The trial will be conducted for the treatment of cognitive impairment in patients with schizophrenia. The primary endpoint is a continuous, normally distributed endpoint (MCCB overall composite score) with a larger value indicating favorable effect. The treatment effect will be evaluated at 3 months after randomization, i.e., the treatment period will 3-month long. A multi-stage adaptive design will be employed in the trial to evaluate the dose-response relationship for the treatment while assigning most patients to the best performing doses. This will be accomplished using a response-adaptive randomization approach.

The following design parameters will be used in this example:

- A total of 320 patients will be enrolled in the trial.
- Patients will be randomly assigned to placebo and four doses of the experimental treatment (20 mg, 50 mg, 100 mg or 250 mg).
- A four-stage design will be utilized with 80 patients enrolled in each stage. A balanced randomization approach will be used in the first stage of the trial and the randomization ratios in the dosing groups will be updated at the end of each stage. The randomization ratio for the placebo group will be fixed at 20% throughout the trial.
- Patients will be enrolled over the period of 24 months with the median enrollment time (time point by which 50% of the patients will be enrolled into the trial) of 18 months.
- The patient dropout rate will be set to 10% at the end of the treatment period.

Dose-response modeling at the end of each stage will be performed using the MCPMod approach based on four dose-response models (linear, exponential, Emax and logistic models). Non-linear parameters of these models will be specified as shown below (note that no parameter needs to be specified for the linear model). The rule for updating the randomization scheme will be based on the posterior probability of achieving a clinically meaningful improvement over placebo in each dosing group. The threshold for clinically meaningful improvement will be set at 1.5.

Operating characteristics of the response-adaptive design will be evaluated using the ADRand function. As shown below, a list of all design and decision rule parameters (`parameters`) will be created

and then passed to this function. It's important to note that, due to the use of Bayesian calculations, the simulation process will be rather slow. It is recommended to take advantage of parallel calculations by setting the number of cores (`ncores`) to a larger value as shown below.

A simulation report for this adaptive design can be created by calling the `GenerateReport` function. The same functionality is also available via a Shiny-based application that can be launched by calling the `ADRandApp` function.

Arguments

No arguments

Value

No return value

See Also

`ADRand`

Examples

```
# List of all parameters

parameters = list()

# Endpoint type
parameters$endpoint_type = "Normal"

# Direction of favorable outcome
parameters$direction = "Higher"

# Dose levels in the trial
parameters$dose_levels = c(0, 20, 50, 100, 250)

# Total number of enrolled patients in each trial stage
parameters$stage_sample_size = c(80, 80, 80, 80)

# Length of the treatment period (months)
parameters$treatment_period = 3

# Length of the patient enrollment period (months)
parameters$enrollment_period = 24

# Median enrollment time (months)
parameters$enrollment_parameter = 18

# Patient dropout rate
parameters$dropout_rate = 0.1

# Mean and SD for the primary endpoint in the placebo group
parameters$control_mean = 1.5
parameters$control_sd = 4
```

```
# Mean and SD for the primary endpoint in the dosing groups
parameters$treatment_mean = c(2, 2.5, 3, 3.5)
parameters$treatment_sd = c(4, 4, 4, 4)

# Fixed randomization ratio in the placebo group
parameters$ratio_placebo = 0.2

# Threshold for clinically meaningful improvement over placebo
parameters$delta = 1.5

# Balance parameter for adaptive randomization
parameters$balance = 2

# Non-linear parameters of the candidate dose-response models used in the MCPMod method
# for modeling the dose-response relationship

# Non-linear parameter of the exponential model (delta)
parameters$exponential_model_parameter = 100

# Non-linear parameter of the Emax model (ED50)
parameters$emax_model_parameter = 200

# Non-linear parameters of the logistic model (ED50 and delta)
parameters$logistic_model_parameters = c(125, 25)

# One-sided Type I error rate
parameters$alpha = 0.025

# Number of cores for parallel calculations
parameters$ncores = 1

# Number of simulations, you should prefer more (use large values!)
parameters$nsims = 10

# Remove this parameter in your code:
parameters$withoutCharts = TRUE

# Run simulations to compute operating characteristics
results = ADRand(parameters)

# Generate a simulation report (remove tempfile)
GenerateReport(results,
               tempfile("ADRand Normally distributed endpoint.docx", fileext=".docx"))
```

ADSSMod *Simulation-based design of adaptive trials with data-driven sample size or event count re-estimation*

Description

This function performs a simulation-based evaluation of operating characteristics for adaptive trials with data-driven sample size re-estimation (normally distributed or binary endpoints) or event count re-estimation (time-to-event endpoint). A two-arm Phase III clinical trial with two interim analyses is assumed. The first interim analysis supports early stopping for futility and the second interim analysis supports an option to increase the number of patients or number of events in the trial. For examples of the function call, see ADSSModExample1, ADSSModExample2 or ADSSModExample3.

Usage

```
ADSSMod(parameters)
```

Arguments

`parameters` List of the trial design and other parameters. The required elements are defined below:

- `endpoint_type`: Character value defining the primary endpoint's type. Possible values:
  - `"Normal"`: Normally distributed endpoint.
  - `"Binary"`: Binary endpoint.
  - `"Time-to-event"`: Time-to-event endpoint.
- `direction`: Character value defining the direction of favorable outcome. Possible values: `"Higher"` (a higher value of the endpoint indicates a more favorable outcome) and `"Lower"` (a lower value of the endpoint indicates a more favorable outcome).
- `sample_size`: Integer vector with two elements defining the number of enrolled patients in the two trial arms (control and experimental treatment). Each element must be positive.
- `control_mean`: Numeric value defining the mean of the primary endpoint in the control arm. This parameter is required only with normally distributed endpoints (`endpoint_type="Normal"`).
- `control_sd`: Numeric value defining the standard deviation of the primary endpoint in the control arm. This value must be positive. This parameter is required only with normally distributed endpoints.
- `treatment_mean`: Numeric value defining the mean of the primary endpoint in the experimental treatment arm. This parameter is required only with normally distributed endpoints.
- `treatment_sd`: Numeric value defining the standard deviation of the primary endpoint in the experimental treatment arm. This value must be positive. This parameter is required only with normally distributed endpoints.
- `control_rate`: Numeric value defining the proportion or response rate for the primary endpoint in the control arm. This value must be between 0 and 1. This parameter is required only with binary endpoints (`endpoint_type="Binary"`).

- `treatment_rate`: Numeric value defining the proportion or response rate for the primary endpoint in the experimental treatment arm. This value must be between 0 and 1. This parameter is required only with binary endpoints.
- `control_time`: Numeric value defining the median time, e.g., the median survival time, for the exponentially distributed primary endpoint in the control arm. This value must be positive. This parameter is required only with time-to-event endpoints (`endpoint_type="Time-to-event"`).
- `treatment_time`: Numeric value defining the median time, e.g., the median survival time, for the exponentially distributed primary endpoint in the experimental treatment arm. This value must be positive. This parameter is required only with time-to-event endpoints.
- `info_frac`: Numeric vector with four elements defining the information fractions at the first interim analysis, at the second interim analysis, at the final analysis before sample size/event count adjustment and at the final analysis after sample size/event count adjustment adjustment. The first and second elements must be between 0 and 1, the third element must be 1 and the fourth element must be greater than 1.
- `event_count`: Numeric value defining the total number of events at the final analysis before event count adjustment. This value must be positive. This parameter is required only with time-to-event endpoints.
- `futility_threshold`: Numeric value defining the futility threshold for conditional power at the first interim analysis. This value must be positive.
- `promising_interval`: Numeric vector with two elements defining the promising interval for conditional power at the second interim analysis. Each element must be between 0 and 1.
- `target_power`: Numeric value defining the target conditional power for increasing the number of patients or events at the second interim analysis. This value must be between 0 and 1.
- `dropout_rate`: Numeric value defining the patient dropout rate. With normally distributed endpoints and binary endpoints, a uniform patient dropout process is assumed and thus this parameter defines the fraction of patients that will be excluded from the interim and final analyses. With time-to-event endpoints, the time to patient dropout is assumed to follow an exponential distribution and this parameter defines the annual dropout rate. This value must be between 0 and 1.
- `enrollment_period`: Numeric value defining the length of the patient enrollment period. This value must be positive. This parameter is required only with time-to-event endpoints.
- `enrollment_parameter`: Numeric value defining the median enrollment time. The patient enrollment process is assumed to be governed by a truncated exponential distribution and this parameter defines the time point by which 50% of the patients are enrolled into the trial. This value must be between 0 and the length of the patient enrollment period. This parameter is required only with time-to-event endpoints..
- alpha: Numeric value defining the overall one-sided Type I error rate. The default value is 0.025.

- `random_seed`: Integer value defining the random number generator seed. The default value is 49283.
- `nsims`: Integer value defining the number of simulation runs.
- `ncores`: Integer value defining the number of cores for parallel calculations. The number of cores cannot exceed the maximum available number of cores. The default value is 1.

Value

The function returns an object of class '`ADSSModResults`'. This object is a list with the following components:

`parameters`	List containing the user-specified parameters.
`sim_results`	Data frame containing key descriptive statistics for each simulation run.
`sim_summary`	List containing the key operating characteristics of the adaptive design.

A detailed summary of the simulation results can be created using the `GenerateReport` function.

See Also

`ADSSModApp`

`ADSSModApp` *Graphical user interface to design an adaptive trial with data-driven sample size or event count re-estimation*

Description

This function creates a Shiny-based graphical user interface to facilitate the process of designing Phase III adaptive trials with two interim analysis to enable a futility assessment and sample size/event count re-estimation. The application supports the same functionality as the `ADSSMod` function. For an example of the function call, see ADSSModExample1.

Usage

```
ADSSModApp()
```

Arguments

No arguments

Value

No return value, called for side effects

See Also

`ADSSMod`

ADSSModExample1 *Simulation-based design of an adaptive trial with sample size re-estimation (normally distributed endpoint)*

Description

Consider a seamless Phase II/Phase III or Phase III trial with a normally distributed primary efficacy endpoint. A larger value of the endpoint corresponds to a more favorable outcome. A single dose or regimen of an experimental treatment will be compared to a control. An adaptive design with two interim analyses will be employed in the trial (the first interim analysis may correspond to the end of the Phase II portion of the trial). The following decision rules will be applied at the interim looks:

- Futility stopping rule will be applied at the first interim analysis (IA1). The trial may be stopped due to futility if a significant treatment effect is unlikely to be established at the final analysis (FA), i.e., the predicted probability of success (conditional power) is very low at this interim analysis.
- Sample size re-estimation rule will be applied at the second interim analysis (IA2). The total number of patients in the trial will be increased to boost the probability of success if the predicted probability of success (conditional power) is lower than expected at this interim analysis.

The following design parameters will be assumed:

- A balanced design with 120 enrolled patients per arm will be utilized.
- The patient dropout rate is equal to 5%, i.e., 5% of the patients are expected to be lost to follow up by the end of the treatment period.
- The information fractions at IA1 and IA2 are set to 40% and 60%, i.e., the first and second interim looks will be taken after 40% and 60% of the patients complete the treatment period or drop out of the trial before completing the treatment period.

In addition, the following decision rules will be considered at the two interim analyses:

- The futility threshold at IA1 is set to 10%, i.e., the trial will be terminated for futility if conditional power does not exceed 10%.
- The promising interval at IA2 ranges from 50% to 90%, i.e., the sample size in the trial will be increased if conditional power is greater than 50% and less than 90%.
- The target conditional power at IA2 is set to 90%, i.e., the sample size will be increased to ensure conditional power of 90% up a pre-defined cap. The maximum increase is equal to 30% of the original sample size.

Finally, the mean effects in the control and treatment arms are assumed to be equal to 0 and 0.3, respectively, with a common standard deviation of 1.

Key operating characteristics of this adaptive design will be evaluated using the `ADSSMod` function with 10,000 simulation runs. To invoke this function, a list of all design and decision rule parameters (`parameters`) needs to be created as shown below. A comprehensive simulation report can be generated by calling the `GenerateReport` function and a graphical user interface can be launched using the `ADSSModApp` function.

Arguments

No arguments

Value

No return value

See Also

ADSSMod, ADSSModExample2, ADSSModExample3

Examples

```
# List of all parameters
parameters = list()

# Endpoint type
parameters$endpoint_type = "Normal"

# Direction of favorable outcome
parameters$direction = "Higher"

# Number of enrolled patients (control, treatment)
parameters$sample_size = c(120, 120)

# Patient dropout rate
parameters$dropout_rate = 0.05

# Mean and SD in the control arm
parameters$control_mean = 0
parameters$control_sd = 1

# Mean and SD in the treatment arm
parameters$treatment_mean = 0.3
parameters$treatment_sd = 1

# Information fractions at IA1, IA2, FA (before sample size adjustment)
# and FA (after sample size adjustment)
parameters$info_frac = c(0.4, 0.6, 1, 1.3)

# Futility threshold for conditional power at IA1
parameters$futility_threshold = 0.1

# Promising interval for conditional power at IA2
parameters$promising_interval = c(0.5, 0.9)

# Target conditional power for increasing the sample size at IA2
parameters$target_power = 0.9

# One-sided alpha level
parameters$alpha = 0.025
```

```
# Number of simulations, you should prefer more
parameters$nsims = 100

# Run simulations to compute operating characteristics
results = ADSSMod(parameters)

# Generate a simulation report (remove tempfile)
GenerateReport(results,
               tempfile("ADSSMod Normally distributed endpoint.docx", fileext=".docx"))
```

ADSSModExample2 *Simulation-based design of an adaptive trial with sample size re-estimation (binary endpoint)*

Description

Consider a seamless Phase II/Phase III or Phase III trial where the primary efficacy endpoint is binary with a higher proportion indicating a more favorable outcome. A single dose or regimen of an experimental treatment will be compared to a control. An adaptive design with two interim analyses will be employed in the trial (the first interim analysis may correspond to the end of the Phase II portion of the trial). The following decision rules will be applied at the interim looks:

- Futility stopping rule will be applied at the first interim analysis (IA1). The trial may be stopped due to futility if a significant treatment effect is unlikely to be established at the final analysis (FA).
- Sample size re-estimation rule will be applied at the second interim analysis (IA2). The total number of patients in the trial will be increased to boost the probability of success if conditional power is lower than expected at this interim analysis.

The following design parameters will be assumed:

- A trial design with a 1:2 randomization ratio will be used. A hundred patients will be enrolled in the control arm and twice as many patients will be enrolled in the experimental treatment arm.
- The patient dropout rate at the end of the treatment period is equal to 10%.
- The information fractions at IA1 and IA2 are set to 40% and 60%.

In addition, the following decision rules will be considered at the two interim analyses:

- The futility threshold at IA1 is set to 20%.
- The promising interval at IA2 ranges from 50% to 90%.
- The target conditional power at IA2 is set to 90% and the total sample size in the trial could be increased up to 40%.

The response rates for the primary efficacy endpoint are assumed to be 10% (control arm)and 25% (experimental treatment arm).

Key operating characteristics of this adaptive design will be evaluated using the ADSSMod function with 10,000 simulation runs. To invoke this function, a list of all design and decision rule parameters

(parameters) needs to be created as shown below. A comprehensive simulation report can be generated by calling the GenerateReport function and a graphical user interface can be launched using the ADSSModApp function.

Arguments

No arguments

Value

No return value

See Also

ADSSMod, ADSSModExample1, ADSSModExample3

Examples

```
# List of all parameters
parameters = list()

# Endpoint type
parameters$endpoint_type = "Binary"

# Direction of favorable outcome
parameters$direction = "Higher"

# Number of enrolled patients (control, treatment)
parameters$sample_size = c(100, 200)

# Patient dropout rate
parameters$dropout_rate = 0.1

# Response rate in the control arm
parameters$control_rate = 0.1

# Response rate in the treatment arm
parameters$treatment_rate = 0.25

# Information fractions at IA1, IA2, FA (before sample size adjustment)
# and FA (after sample size adjustment)
parameters$info_frac = c(0.4, 0.6, 1, 1.4)

# Futility threshold for conditional power at IA1
parameters$futility_threshold = 0.2

# Promising interval for conditional power at IA2
parameters$promising_interval = c(0.5, 0.9)

# Target conditional power for increasing the sample size at IA2
parameters$target_power = 0.9
```

```
# One-sided alpha level
parameters$alpha = 0.025

# Number of simulations, you should prefer more
parameters$nsims = 100

# Run simulations to compute operating characteristics
results = ADSSMod(parameters)

# Generate a simulation report (remove tempfile)
GenerateReport(results,
               tempfile("ADSSMod Binary endpoint.docx", fileext=".docx"))
```

ADSSModExample3 *Simulation-based design of an adaptive trial with event count re-estimation (time-to-event endpoint)*

Description

Consider a seamless Phase II/Phase III or Phase III trial where the primary efficacy endpoint is a time-to-event endpoint, e.g., the time to disease progression or death. A single regimen of an experimental treatment will be compared to a control. An adaptive design with two interim analyses will be employed in the trial (the first interim analysis may correspond to the end of the Phase II portion of the trial). The following decision rules will be applied at the interim looks:

- Futility stopping rule will be applied at the first interim analysis (IA1). The trial may be stopped due to futility if a significant treatment effect is unlikely to be established at the final analysis (FA).
- Event count re-estimation rule will be applied at the second interim analysis (IA2). The target number of events of interest will be increased to improve the probability of success if conditional power is lower than expected at this interim analysis.

The following design parameters will be assumed:

- A balanced design with 220 enrolled patients per arm will be used in the trial.
- The original number of events at FA is set to 300.
- The annual patient dropout rate is equal to 5% (the time to patient dropout is assumed to follow an exponential distribution).
- The length of the patient enrollment period is 12 months and the median enrollment time is anticipated to be 8 months, i.e., 50% of the patients are expected to be enrolled into the trial by the 8-month milestone.
- The information fractions at IA1 and IA2 are set to 40% and 60%, i.e., the first and second interim analyses will be performed after 40% and 60% of the original number of events have been accrued.

In addition, the following decision rules will be considered at the two interim analyses:

- The futility threshold at IA1 is set to 10%.

- The promising interval at IA2 ranges from 50% to 90%.
- The target conditional power at IA2 is set to 90% and the target number of events could be increased up to 30%.

The median event times, e.g., median survival times, are assumed to be 7.5 months and 10.5 months in the control and experimental treatment arms, respectively. If the time to the event of interest is exponentially distributed, these assumptions correspond to a hazard ratio of 0.71.

Key operating characteristics of this adaptive design will be evaluated using the ADSSMod function with 10,000 simulation runs. To invoke this function, a list of all design and decision rule parameters (parameters) needs to be created as shown below. A comprehensive simulation report can be generated by calling the GenerateReport function and a graphical user interface can be launched using the ADSSModApp function.

Arguments

No arguments

Value

No return value

See Also

ADSSMod, ADSSModExample1, ADSSModExample2

Examples

```
# List of all parameters
parameters = list()

# Endpoint type
parameters$endpoint_type = "Time-to-event"

# Direction of favorable outcome
parameters$direction = "Higher"

# Number of enrolled patients (control, treatment)
parameters$sample_size = c(220, 220)

# Annual patient dropout rate
parameters$dropout_rate = 0.05

# Median time in the control arm
parameters$control_time = 7.5

# Median time in the treatment arm
parameters$treatment_time = 10.5

# Target event count at FA (before event count adjustment)
parameters$event_count = 300
```

```
# Information fractions at IA1, IA2, FA (before event count adjustment)
# and FA (after event count adjustment)
parameters$info_frac = c(0.4, 0.6, 1, 1.3)

# Futility threshold for conditional power at IA1
parameters$futility_threshold = 0.1

# Promising interval for conditional power at IA2
parameters$promising_interval = c(0.5, 0.9)

# Target conditional power for increasing the number of events at IA2
parameters$target_power = 0.9

# Enrollment period
parameters$enrollment_period = 12

# Median enrollment time
parameters$enrollment_parameter = 8

# One-sided alpha level
parameters$alpha = 0.025

# Number of simulations, you should prefer more
parameters$nsims = 100

# Run simulations to compute operating characteristics
results = ADSSMod(parameters)

# Generate a simulation report (remove tempfile)
GenerateReport(results,
               tempfile("ADSSMod Time-to-event endpoint.docx", fileext=".docx"))
```

`ADTreatSel` *Simulation-based design of adaptive trials with data-driven treatment selection*

Description

This function performs a simulation-based evaluation of operating characteristics for adaptive trials with data-driven treatment selection. A multi-arm Phase III clinical trial with several doses or regimens of an experimental treatment versus control and two interim analyses is assumed. The first interim analysis supports early stopping for futility and the second interim analysis enables adaptive treatment selection to identify the best performing treatment. For examples of the function call, see ADTreatSelExample1, ADTreatSelExample2 or ADTreatSelExample3.

Usage

```
ADTreatSel(parameters)
```

Arguments

`parameters` List of the trial design and other parameters. The required elements are defined below:

- `endpoint_type`: Character value defining the primary endpoint's type. Possible values:
  - `"Normal"`: Normally distributed endpoint.
  - `"Binary"`: Binary endpoint.
  - `"Time-to-event"`: Time-to-event endpoint.
- `direction`: Character value defining the direction of favorable outcome. Possible values: `"Higher"` (a higher value of the endpoint indicates a more favorable outcome) and `"Lower"` (a lower value of the endpoint indicates a more favorable outcome).
- `sample_size`: Integer vector defining the number of enrolled patients in the trial arms (control and multiple experimental treatments). Each element must be positive.
- `control_mean`: Numeric value defining the mean of the primary endpoint in the control arm. This parameter is required only with normally distributed endpoints (`endpoint_type="Normal"`).
- `control_sd`: Numeric value defining the standard deviation of the primary endpoint in the control arm. This value must be positive. This parameter is required only with normally distributed endpoints.
- `treatment_mean`: Numeric vector defining the means of the primary endpoint in the experimental treatment arms. This parameter is required only with normally distributed endpoints.
- `treatment_sd`: Numeric vector defining the standard deviations of the primary endpoint in the experimental treatment arms. Each element must be positive. This parameter is required only with normally distributed endpoints.
- `control_rate`: Numeric value defining the proportion or response rate for the primary endpoint in the control arm. This value must be between 0 and 1. This parameter is required only with binary endpoints (`endpoint_type="Binary"`).
- `treatment_rate`: Numeric vector defining the proportions or response rates for the primary endpoint in the experimental treatment arms. Each element must be between 0 and 1. This parameter is required only with binary endpoints.
- `control_time`: Numeric value defining the median time, e.g., the median survival time, for the exponentially distributed primary endpoint in the control arm. This value must be positive. This parameter is required only with time-to-event endpoints (`endpoint_type="Time-to-event"`).
- `treatment_time`: Numeric vector defining the median times, e.g., the median survival times, for the exponentially distributed primary endpoint in the experimental treatment arms. Each element must be positive. This parameter is required only with time-to-event endpoints.
- `info_frac`: Numeric vector with three elements defining the information fractions at the first interim analysis, at the second interim analysis and at

the final analysis. The first and second elements must be between 0 and 1, and the third element must be 1.

- `event_count`: Numeric value defining the total number of events at the final analysis. This value must be positive. This parameter is required only with time-to-event endpoints.
- `futility_threshold`: Numeric value defining the futility threshold for conditional power at the first interim analysis. This value must be positive.
- `dropout_rate`: Numeric value defining the patient dropout rate. With normally distributed endpoints and binary endpoints, a uniform patient dropout process is assumed and thus this parameter defines the fraction of patients that will be excluded from the interim and final analyses. With time-to-event endpoints, the time to patient dropout is assumed to follow an exponential distribution and this parameter defines the annual dropout rate. This value must be between 0 and 1.
- `enrollment_period`: Numeric value defining the length of the patient enrollment period. This value must be positive. This parameter is required only with time-to-event endpoints.
- `enrollment_parameter`: Numeric value defining the median enrollment time. The patient enrollment process is assumed to be governed by a truncated exponential distribution and this parameter defines the time point by which 50% of the patients are enrolled into the trial. This value must be between 0 and the length of the patient enrollment period. This parameter is required only with time-to-event endpoints..
- `treatment_count`: Integer value defining the number of treatments to be selected at the second interim analysis. This value must range between 1 and the total number of treatments in the trial.
- `mult_test`: Character value defining the multiple testing procedure to be used in the trial. Possible values:
  - `"Bonferroni"`: Bonferroni multiple testing procedure.
  - `"Holm"`: Holm multiple testing procedure.
  - `"Hochberg"`: Hochberg multiple testing procedure.

  `alpha`: Numeric value defining the overall one-sided Type I error rate. The default value is 0.025. `random_seed`: Integer value defining the random number generator seed. The default value is 49283. `nsims`: Integer value defining the number of simulation runs. `ncores`: Integer value defining the number of cores for parallel calculations. The number of cores cannot exceed the maximum available number of cores. The default value is 1.

Value

The function returns an object of class '`ADTreatSelResults`'. This object is a list with the following components:

`parameters`	List containing the user-specified parameters.
`sim_results`	Data frame containing key descriptive statistics for each simulation run.
`sim_summary`	List containing the key operating characteristics of the adaptive design.

A detailed summary of the simulation results can be created using the `GenerateReport` function.

See Also

ADTreatSelApp

ADTreatSelApp *Graphical user interface to design an adaptive trial with data-driven treatment selection*

Description

This function creates a web application with a Shiny-based graphical user interface to facilitate the process of designing Phase III adaptive trials with two interim analysis to enable a futility assessment and data-driven selection of the best treatment arm. The application supports the same functionality as the ADTreatSel function. For an example of the function call, see ADTreatSelExample1.

Usage

```
ADTreatSelApp()
```

Arguments

No arguments

Value

No return value, called for side effects

See Also

ADTreatSel

ADTreatSelExample1 *Simulation-based design of an adaptive trial with treatment selection (normally distributed endpoint)*

Description

Consider a seamless Phase II/Phase III or Phase III trial with a normally distributed primary efficacy endpoint with a larger value indicating a more favorable outcome. The efficacy and safety profiles of two treatments, e.g., two doses of an experimental treatment, will be evaluated in this trial compared to a control. An adaptive design with two interim analyses will be employed in the trial (the first interim analysis may correspond to the end of the Phase II portion of the trial). The following decision rules will be applied at the interim looks:

- Futility stopping rule will be applied at the first interim analysis (IA1). A treatment arm may be dropped due to futility if the treatment-specific conditional power is too low. The trial will be terminated for futility if all treatment arms are dropped.

- Treatment selection rule will be applied at the second interim analysis (IA2). The best performing treatment will be selected for the final analysis (FA) and the remaining treatment will be dropped at this interim analysis.

The following design parameters will be assumed:

- A balanced design with 200 enrolled patients per trial arm will be utilized in the trial.
- The patient dropout rate at the end of the treatment period is equal to 5%.
- The information fractions at IA1 and IA2 are set to 40% and 60%, i.e., the first and second interim looks will be taken after 40% and 60% of the patients complete the treatment period or drop out of the trial before completing the treatment period.

The futility threshold at IA1 is set to 25%, which means that a treatment arm will be dropped at this interim analysis if conditional power does not exceed 25%. The mean effect and standard deviation are assumed to be 0 and 1 in the control arm. The mean effects in the two treatment arms are assumed to be equal to 0.25 and 0.3, respectively, with a common standard deviation of 1.

To address multiplicity induced by the data-driven treatment selection, the Hochberg procedure will be applied at the final analysis.

Key operating characteristics of this adaptive design with a treatment selection option at the second interim look will be evaluated using the `ADTreatSel` function based on 10,000 simulation runs. To invoke this function, a list of all design and decision rule parameters (`parameters`) needs to be created as shown below. A comprehensive simulation report for this adaptive design can be generated by calling the `GenerateReport` function and a graphical user interface can be launched using the `ADTreatSelApp` function.

Arguments

No arguments

Value

No return value

See Also

`ADTreatSel`, `ADTreatSelExample2`, `ADTreatSelExample3`

Examples

```
# List of all parameters
parameters = list()

# Endpoint type
parameters$endpoint_type = "Normal"

# Direction of favorable outcome
parameters$direction = "Higher"

# Number of enrolled patients (control, two treatments)
parameters$sample_size = c(200, 200, 200)
```

```
# Patient dropout rate
parameters$dropout_rate = 0.05

# Mean and SD in the control arm
parameters$control_mean = 0
parameters$control_sd = 1

# Means and SDs in the multiple treatment arms
parameters$treatment_mean = c(0.25, 0.30)
parameters$treatment_sd = c(1, 1)

# Information fractions at IA1, IA2, FA
parameters$info_frac = c(0.4, 0.6, 1)

# Futility threshold for conditional power at IA1
parameters$futility_threshold = 0.25

# Number of treatments to be selected at the second interim analysis
parameters$treatment_count = 1

# Multiple testing procedure to be used in the trial
parameters$mult_test = "Hochberg"

# One-sided alpha level
parameters$alpha = 0.025

# Number of simulations, you should prefer more
parameters$nsims = 100

# Run simulations to compute operating characteristics
results = ADTreatSel(parameters)

# Generate a simulation report (remove tempfile)
GenerateReport(results,
             tempfile("ADTreatSel Normally distributed endpoint.docx", fileext=".docx"))
```

`ADTreatSelExample2` *Simulation-based design of an adaptive trial with treatment selection (binary endpoint)*

Description

Consider a seamless Phase II/Phase III or Phase III trial with a binary primary efficacy endpoint. This endpoint represents a success rate and thus a higher proportion corresponds to a more favorable outcome. The efficacy and safety profiles of three treatments, e.g., three doses of an experimental treatment, will be evaluated in this trial compared to a control. An adaptive design with two interim analyses will be employed in the trial (the first interim analysis may correspond to the end of the Phase II portion of the trial). The following decision rules will be applied at the interim looks:

- Futility stopping rule will be applied at the first interim analysis (IA1). A treatment arm may be dropped due to futility if the treatment-specific conditional power is too low. The trial will be terminated for futility if all treatment arms are dropped.
- Treatment selection rule will be applied at the second interim analysis (IA2). Two best performing treatments will be selected for the final analysis (FA) and patient enrollment will be discontinued in the third treatment arm after the second interim analysis.

The following design parameters will be assumed:

- A balanced design with 150 enrolled patients per trial arm will be utilized in the trial.
- The patient dropout rate at the end of the treatment period is equal to 10%.
- The information fractions at IA1 and IA2 are set to 40% and 60%.

The futility threshold for each treatment arm at IA1 is set to 15%. The response rates for the primary endpoint are assumed to be 10% in the control arm and 25% in each treatment arm.

To protect the overall Type I error rate in the trial, the Hochberg procedure will be used at the final analysis.

Key operating characteristics of this adaptive design with a treatment selection option at the second interim look will be evaluated using the `ADTreatSel` function based on 10,000 simulation runs. To invoke this function, a list of all design and decision rule parameters (`parameters`) needs to be created as shown below. A comprehensive simulation report for this adaptive design can be generated by calling the `GenerateReport` function and a graphical user interface can be launched using the `ADTreatSelApp` function.

Arguments

No arguments

Value

No return value

See Also

`ADTreatSel`, `ADTreatSelExample1`, `ADTreatSelExample3`

Examples

```
# List of all parameters
parameters = list()

# Endpoint type
parameters$endpoint_type = "Binary"

# Direction of favorable outcome
parameters$direction = "Higher"

# Number of enrolled patients (control, three treatments)
parameters$sample_size = c(150, 150, 150, 150)
```

```
# Patient dropout rate
parameters$dropout_rate = 0.1

# Response rate in the control arm
parameters$control_rate = 0.1

# Response rates in the treatment arms
parameters$treatment_rate = c(0.25, 0.25, 0.25)

# Information fractions at IA1, IA2, FA
parameters$info_frac = c(0.4, 0.6, 1)

# Futility threshold for conditional power at IA1
parameters$futility_threshold = 0.15

# Number of treatments to be selected at the second interim analysis
parameters$treatment_count = 2

# Multiple testing procedure to be used in the trial
parameters$mult_test = "Hochberg"

# One-sided alpha level
parameters$alpha = 0.025

# Number of simulations, you should prefer more
parameters$nsims = 100

# Run simulations to compute operating characteristics
results = ADTreatSel(parameters)

# Generate a simulation report (remove tempfile)
GenerateReport(results,
               tempfile("ADTreatSel Binary endpoint.docx", fileext=".docx"))
```

`ADTreatSelExample3` *Simulation-based design of an adaptive trial with treatment selection (time-to-event endpoint)*

Description

Consider a seamless Phase II/Phase III or Phase III trial with a time-to-event primary efficacy endpoint. Two regimens of an experimental treatment will be evaluated in this trial compared to a control. An adaptive design with two interim analyses will be employed in the trial (the first interim analysis may correspond to the end of the Phase II portion of the trial). The following decision rules will be applied at the interim looks:

- Futility stopping rule will be applied at the first interim analysis (IA1). A treatment arm may be dropped due to futility if the treatment-specific conditional power is too low. The trial will be terminated for futility if all treatment arms are dropped.

- Treatment selection rule will be applied at the second interim analysis (IA2). The best performing treatment will be selected for the final analysis (FA) and the other treatment will be dropped at this interim analysis.

The following design parameters will be assumed:

- A balanced design with 180 enrolled patients per arm will be used in the trial.
- The target number of events at FA is set to 450.
- The annual patient dropout rate is equal to 5% (the time to patient dropout is assumed to follow an exponential distribution).
- The length of the patient enrollment period is 24 months and the median enrollment time is expected to be 18 months.
- The information fractions at IA1 and IA2 are set to 40% and 60%, i.e., the first and second interim analyses will be performed after 40% and 60% of the original number of events have been accrued.

The futility threshold for each treatment arm at IA1 is set to 10%. The median event times are assumed to be 7.5 months in the control arms and 10.5 months in both treatment arms, respectively. Under the assumption of exponentially distributed event times, these assumptions correspond to a hazard ratio of 0.71 in each treatment arm.

To control the overall Type I error rate in the trial, the Hochberg procedure will be applied at the final analysis.

Key operating characteristics of this adaptive design with a treatment selection option at the second interim look will be evaluated using the `ADTreatSel` function based on 10,000 simulation runs. To invoke this function, a list of all design and decision rule parameters (`parameters`) needs to be created as shown below. A comprehensive simulation report for this adaptive design can be generated by calling the `GenerateReport` function and a graphical user interface can be launched using the `ADTreatSelApp` function.

Arguments

No arguments

Value

No return value

See Also

`ADTreatSel`, `ADTreatSelExample1`, `ADTreatSelExample2`

Examples

```
# List of all parameters
parameters = list()

# Endpoint type
parameters$endpoint_type = "Time-to-event"
```

```
# Direction of favorable outcome
parameters$direction = "Higher"

# Number of enrolled patients (control, two treatments)
parameters$sample_size = c(180, 180, 180)

# Annual patient dropout rate
parameters$dropout_rate = 0.05

# Median time in the control arm
parameters$control_time = 7.5

# Median times in the treatment arms
parameters$treatment_time = c(10.5, 10.5)

# Target event count at FA
parameters$event_count = 450

# Information fractions at IA1, IA2, FA
parameters$info_frac = c(0.4, 0.6, 1)

# Futility threshold for conditional power at IA1
parameters$futility_threshold = 0.1

# Enrollment period
parameters$enrollment_period = 24

# Median enrollment time
parameters$enrollment_parameter = 18

# Number of treatments to be selected at the second interim analysis
parameters$treatment_count = 1

# Multiple testing procedure to be used in the trial
parameters$mult_test = "Hochberg"

# One-sided alpha level
parameters$alpha = 0.025

# Number of simulations, you should prefer more
parameters$nsims = 100

# Run simulations to compute operating characteristics
results = ADTreatSel(parameters)

# Generate a simulation report (remove tempfile)
GenerateReport(results,
               tempfile("ADTreatSel Time-to-event endpoint.docx", fileext=".docx"))
```

ClustRand *Simulation-based design of cluster-randomized trials*

Description

This function performs a simulation-based evaluation of operating characteristics for traditional cluster-randomized trials with normally distributed or binary endpoints. A two-arm clinical trial is assumed (treatment and control arms). For examples of the function call, see ClustRandExample1 or ClustRandExample2.

Usage

```
ClustRand(parameters)
```

Arguments

`parameters` List of the trial design and other parameters. The required elements are defined below:

- `endpoint_type`: Character value defining the primary endpoint's type. Possible values:
  - `"Normal"`: Normally distributed endpoint.
  - `"Binary"`: Binary endpoint.
- `direction`: Character value defining the direction of favorable outcome. Possible values: `"Higher"` (a higher value of the endpoint indicates a more favorable outcome) and `"Lower"` (a lower value of the endpoint indicates a more favorable outcome).
- `sample_size`: Integer vector with two elements defining the number of completers in the two trial arms (control and experimental treatment). Completers are defined as patients who complete the trial and are included in the final analysis. Each element must be positive.
- `cluster_scheme`: Character value defining the cluster scheme. Possible values:
  - `"Fixed"`: Cluster sizes are pre-defined.
  - `"Random"`: Cluster sizes are randomly generated.
- `control_cluster_size`: Numeric vector defining the number of cluster sizes in the control arm. This parameter is required only if the cluster sizes are pre-defined (`cluster_scheme="Fixed"`).
- `treatment_cluster_size`: Numeric vector defining the number of cluster sizes in the treatment arm. This parameter is required only if the cluster sizes are pre-defined (`cluster_scheme="Fixed"`).
- `n_clusters`: Numeric vector with two elements defining the number of clusters in the two trial arms (control and experimental treatment). The minimum cluster size (5 patients per cluster) is required and, as a result, with random cluster sizes, the total sample size in a trial arm may exceed the target sample size if a large number of small clusters are generated. This parameter is required only if the cluster sizes are randomly generated (`cluster_scheme="Random"`).
- `cluster_cv`: Numeric value defining the coefficient of variation for the cluster size. This value must be between 0 and 0.5. This parameter is required only if the cluster sizes are randomly generated (`cluster_scheme="Random"`).

- `control_mean`: Numeric value defining the mean of the primary endpoint in the control arm. This parameter is required only with normally distributed endpoints (`endpoint_type="Normal"`).
- `treatment_mean`: Numeric value defining the mean of the primary endpoint in the experimental treatment arm. This parameter is required only with normally distributed endpoints.
- `control_rate`: Numeric value defining the proportion or response rate for the primary endpoint in the control arm. This value must be between 0 and 1. This parameter is required only with binary endpoints (`endpoint_type="Binary"`).
- `treatment_rate`: Numeric value defining the proportion or response rate for the primary endpoint in the experimental treatment arm. This value must be between 0 and 1. This parameter is required only with binary endpoints.
- `control_icc`: Numeric value defining the intra-cluster correlation coefficient in the control arm. This value must be between 0 and 1.
- `control_between_cluster_sd`: Numeric value defining the between-cluster standard deviation in the control arm. This value must be positive.
- `treatment_icc`: Numeric value defining the intra-cluster correlation coefficient in the treatment arm. This value must be between 0 and 1.
- `treatment_between_cluster_sd`: Numeric value defining the between-cluster standard deviation in the treatment arm. This value must be positive.
- alpha: Numeric value defining the overall one-sided Type I error rate. The default value is 0.025.
- `random_seed`: Integer value defining the random number generator seed. The default value is 49283.
- `nsims`: Integer value defining the number of simulation runs.
- `ncores`: Integer value defining the number of cores for parallel calculations. The number of cores cannot exceed the maximum available number of cores. The default value is 1.

Value

The function returns an object of class '`ClustRandResults`'. This object is a list with the following components:

`parameters`	List containing the user-specified parameters.
`sim_results`	Data frame containing key descriptive statistics for each simulation run.
`sim_summary`	List containing the key operating characteristics of the adaptive design.

A detailed summary of the simulation results can be created using the `GenerateReport` function.

See Also

`ClustRandApp`

`ClustRandApp` *Graphical user interface to design a cluster-randomized trial*

Description

This function creates a web application with a Shiny-based graphical user interface to facilitate the process of designing a two-arm cluster-randomized trial (treatment and control arms). The application supports the same functionality as the `ClustRand` function. For an example of the function call, see ClustRandExample1.

Usage

```
ClustRandApp()
```

Arguments

No arguments

Value

No return value, called for side effects

See Also

`ClustRand`

`ClustRandExample1` *Simulation-based design of a cluster-randomized trial (normally distributed endpoint)*

Description

Consider a seamless Phase II/Phase III or Phase III trial with a normally distributed primary efficacy endpoint. A larger value of the endpoint corresponds to a more favorable outcome. A single dose or regimen of an experimental treatment will be compared to a control. An adaptive design with two interim analyses will be employed in the trial (the first interim analysis may correspond to the end of the Phase II portion of the trial). The following decision rules will be applied at the interim looks:

- Futility stopping rule will be applied at the first interim analysis (IA1). The trial may be stopped due to futility if a significant treatment effect is unlikely to be established at the final analysis (FA), i.e., the predicted probability of success (conditional power) is very low at this interim analysis.
- Sample size re-estimation rule will be applied at the second interim analysis (IA2). The total number of patients in the trial will be increased to boost the probability of success if the predicted probability of success (conditional power) is lower than expected at this interim analysis.

The following design parameters will be assumed:

- A balanced design with 120 enrolled patients per arm will be utilized.
- The patient dropout rate is equal to 5%, i.e., 5% of the patients are expected to be lost to follow up by the end of the treatment period.
- The information fractions at IA1 and IA2 are set to 40% and 60%, i.e., the first and second interim looks will be taken after 40% and 60% of the patients complete the treatment period or drop out of the trial before completing the treatment period.

In addition, the following decision rules will be considered at the two interim analyses:

- The futility threshold at IA1 is set to 10%, i.e., the trial will be terminated for futility if conditional power does not exceed 10%.
- The promising interval at IA2 ranges from 50% to 90%, i.e., the sample size in the trial will be increased if conditional power is greater than 50% and less than 90%.
- The target conditional power at IA2 is set to 90%, i.e., the sample size will be increased to ensure conditional power of 90% up a pre-defined cap. The maximum increase is equal to 30% of the original sample size.

Finally, the mean effects in the control and treatment arms are assumed to be equal to 0 and 0.3, respectively, with a common standard deviation of 1.

Key operating characteristics of this adaptive design will be evaluated using the `ClustRand` function with 10,000 simulation runs. To invoke this function, a list of all design and decision rule parameters (`parameters`) needs to be created as shown below. A comprehensive simulation report can be generated by calling the `GenerateReport` function and a graphical user interface can be launched using the `ClustRandApp` function.

Arguments

No arguments

Value

No return value

See Also

`ClustRand`, `ClustRandExample2`

Examples

```
# List of all parameters
parameters = list()

# Endpoint type
parameters$endpoint_type = "Normal"

# Direction of favorable outcome
parameters$direction = "Lower"
```

```
# Number of completers in the trial arms (control, multiple treatments)
parameters$sample_size = c(100, 100)

# Cluster scheme
parameters$cluster_scheme = "Fixed"

# Vector of cluster sizes in the control arm
parameters$control_cluster_size = rep(10, 10)

# Vector of cluster sizes in the treatment arm
parameters$treatment_cluster_size = rep(10, 10)

# Mean in the control arm
parameters$control_mean = 1

# Intra-cluster correlation coefficient in the control arm
parameters$control_icc = 0.6

# Between-cluster standard deviation in the control arm
parameters$control_between_cluster_sd = 1.2

# Mean in the treatment arm
parameters$treatment_mean = 0.3

# Intra-cluster correlation coefficient in the treatment arm
parameters$treatment_icc = 0.6

# Between-cluster standard deviation in the treatment arm
parameters$treatment_between_cluster_sd = 1.2

# Data analysis method (generalized estimating equations (GEE)
# or generalized linear mixed effects model (GLMEM))
parameters$method_type = "GEE"

# One-sided alpha level
parameters$alpha = 0.025

# Number of simulations
parameters$nsims = 10

# Number of cores for parallel calculations
parameters$ncores = 1

# Compute descriptive statistics (arm-specific effects, ICC, cluster sizes) for each simulation run
parameters$descriptive_statistics = TRUE

# Run simulations to compute operating characteristics
results = ClustRand(parameters)

# Generate a simulation report (remove tempfile)
GenerateReport(results,
               tempfile("ClustRand Normally distributed endpoint.docx", fileext=".docx"))
```

`ClustRandExample2` *Simulation-based design of a cluster-randomized trial (binary endpoint)*

Description

Consider a seamless Phase II/Phase III or Phase III trial where the primary efficacy endpoint is binary with a higher proportion indicating a more favorable outcome. A single dose or regimen of an experimental treatment will be compared to a control. An adaptive design with two interim analyses will be employed in the trial (the first interim analysis may correspond to the end of the Phase II portion of the trial). The following decision rules will be applied at the interim looks:

- Futility stopping rule will be applied at the first interim analysis (IA1). The trial may be stopped due to futility if a significant treatment effect is unlikely to be established at the final analysis (FA).
- Sample size re-estimation rule will be applied at the second interim analysis (IA2). The total number of patients in the trial will be increased to boost the probability of success if conditional power is lower than expected at this interim analysis.

The following design parameters will be assumed:

- A trial design with a 1:2 randomization ratio will be used. A hundred patients will be enrolled in the control arm and twice as many patients will be enrolled in the experimental treatment arm.
- The patient dropout rate at the end of the treatment period is equal to 10%.
- The information fractions at IA1 and IA2 are set to 40% and 60%.

In addition, the following decision rules will be considered at the two interim analyses:

- The futility threshold at IA1 is set to 20%.
- The promising interval at IA2 ranges from 50% to 90%.
- The target conditional power at IA2 is set to 90% and the total sample size in the trial could be increased up to 40%.

The response rates for the primary efficacy endpoint are assumed to be 10% (control arm)and 25% (experimental treatment arm).

Key operating characteristics of this adaptive design will be evaluated using the `ClustRand` function with 10,000 simulation runs. To invoke this function, a list of all design and decision rule parameters (`parameters`) needs to be created as shown below. A comprehensive simulation report can be generated by calling the `GenerateReport` function and a graphical user interface can be launched using the `ClustRandApp` function.

Arguments

No arguments

Value

No return value

See Also

ClustRand, ClustRandExample1

Examples

```
# List of all parameters
parameters = list()

# Endpoint type
parameters$endpoint_type = "Binary"

# Direction of favorable outcome
parameters$direction = "Lower"

# Number of completers in the trial arms (control, multiple treatments)
parameters$sample_size = c(100, 100)

# Cluster scheme (fixed or random cluster sizes)
parameters$cluster_scheme = "Fixed"

# Vector of cluster sizes in the control arm
parameters$control_cluster_size = rep(10, 10)

# Vector of cluster sizes in the treatment arm
parameters$treatment_cluster_size = rep(10, 10)

# Response rate in the control arm
parameters$control_rate = 0.6

# Intracluster correlation coefficient in the control arm
parameters$control_icc = 0.3

# Response rate in the treatment arms
parameters$treatment_rate = 0.3

# Intracluster correlation coefficient in the treatment arms
parameters$treatment_icc = 0.3

# Data analysis method (generalized estimating equations (GEE)
# or generalized linear mixed effects model (GLMEM))
parameters$method_type = "GLMEM"

# One-sided alpha level
parameters$alpha = 0.025

# Number of simulations
parameters$nsims = 10
```

```
# Number of cores for parallel calculations
parameters$ncores = 1

# Run simulations to compute operating characteristics
results = ClustRand(parameters)

# Generate a simulation report (remove tempfile)
GenerateReport(results,
               tempfile("ClustRand Binary endpoint.docx", fileext=".docx"))
```

`EventPred` *Simulation-based event prediction in trials with an event-driven design*

Description

This function performs event forecasting in trials with an event-driven design. Blinded event data at an interim analysis are used to forecast the number of events at pre-defined time points in the future. For an example of the function call, see EventPredExample.

Usage

```
EventPred(parameters)
```

Arguments

`parameters` List of the trial design and other parameters. The required elements are defined below:

- `data_set`: Data frame that contains blinded patient enrollment, event and dropout data at the interim analysis with a single record per patient. The following :
  - `enrollment`: Time of the patient's enrollment.
  - `time`: Time of the patient's death or last contact if the event is censored relative to the patient's enrollment.
  - `event`: Event indicator (1 if the patient died and 0 otherwise).
  - `dropout`: Patient dropout indicator (1 if the patient dropped out of the trial/was lost to follow up and 0 otherwise).

  An example is provided in the `EventPredData` data set.
- `time_points`: Numeric vector defining the future time points for computing event predictions. Each elements must be greater than the latest time point in the `data_set`.
- `event_prior_distribution`: Numeric vector with two elements defining the shape and rate parameters of the prior gamma distribution for the event hazard rate. Each element must be positive. The parameters could be computed using the `EventPredPriorDistribution` function.

- dropout_prior_distribution: Numeric vector with two elements defining the shape and rate parameters of the prior gamma distribution for the patient dropout hazard rate. Each element must be positive. The parameters could be computed using the EventPredPriorDistribution function.
- enrollment_prior_distribution: Numeric vector with two elements defining the shape and rate parameters of the prior gamma distribution for the patient enrollment rate. Each element must be positive. The parameters could be computed using the EventPredPriorDistribution function.
- random_seed: Integer value defining the random number generator seed. The default value is 49283.
- nsims: Integer value defining the number of simulation runs.
- ncores: Integer value defining the number of cores for parallel calculations. The number of cores cannot exceed the maximum available number of cores. The default value is 1.

Value

The function returns an object of class 'EventPredResults'. This object is a list with the following components:

parameters — List containing the user-specified parameters.

interim_analysis — Data frame containing the event information from the interim analysis data set.

predictions — Data frame containing the event prediction information at the future time points.

A detailed summary of the simulation results can be created using the GenerateReport function.

See Also

EventPredApp

EventPredApp *Graphical user interface for event prediction in trials with an event-driven design*

Description

This function creates a web application with a Shiny-based graphical user interface to facilitate the process of event forecasting in a Phase II or Phase III clinical trial with an event-driven design. The application supports the same functionality as the EventPred function. For an example of the function call, see EventPredExample.

This application prompts the user to upload a data set in a CSV (comma-separated values) format with blinded patient enrollment, event and patient dropout data. The data set is required to include four variables (enrollment, time, event and dropout) with a single record per patient, see for example the EventPredData data set.

Usage

```
EventPredApp()
```

Arguments

No arguments

Value

No return value, called for side effects

See Also

EventPred

`EventPredData` *Example data set for EventPred*

Description

Example data set based on a Phase III oncology trial with an overall survival endpoint to illustrate event prediction in event-driven trials.

Usage

```
data(EventPredData)
```

Arguments

No arguments

Format

A data set with 150 records (one record per patient) and 4 variables:

`enrollment` Time of the patient's enrollment (months).

`time` Time of the patient's death or last contact if the event is censored relative to the patient's enrollment (months).

`event` Event indicator (1 if the patient died and 0 otherwise).

`dropout` Patient dropout indicator (1 if the patient dropped out of the trial/was lost to follow up and 0 otherwise).

Value

No return value

EventPredExample *Simulation-based event prediction in trials with an event-driven design (time-to- event endpoint)*

Description

Consider a Phase III trial in an oncology population that is being conducted to investigate the efficacy of an experimental treatment versus a control (best supportive care). The primary efficacy analysis evaluates the survival benefit of this treatment. Blinded data, including patient enrollment, event and patient dropout information, will be reviewed periodically throughout the trial to predict the number of events at pre-defined time points in the future.

Assume that the current snapshot of data required for event prediction is included in the EventPredData data set. This data set contains blinded patient enrollment, event and patient dropout information on 150 patients at 12 months after the study start. The blinded data, along with relevant prior information, will support the calculation of the predicted number of events and 95% predictive intervals up to the 24-month milestone.

Prior information needs to be provided for the event hazard rate, patient dropout hazard rate and patient enrollment rate. These parameters are assumed to follow gamma distributions. The shape and rate parameters of the gamma distributions could be specified directly or by invoking the EventPredPriorDistribution function. This function facilitates the process of prior specifications by allowing the user to provide the expected value of a parameter of interest, e.g., the event hazard rate, as well as the uncertainty parameter. The latter quantifies the amount of variability around the expected value, e.g., the uncertainty parameter of 0.1 corresponds to a low standard deviation and thus defines a high-confidence prior distribution. The expected values of the event and patient enrollment hazard rates will be computed using the assumptions that the expected median event and dropout times are equal to 15 and 80 months, respectively (the expected median dropout time corresponds to an annual dropout rate of 10%). The expected patient enrollment rate will be set to 35 patients per month. A common uncertainty parameter of 0.3 will be used for the the event hazard rate, patient dropout hazard rate and patient enrollment rate. This uncertainty parameter will define medium-confidence prior distributions for the three parameters of interest.

A summary of event predictions can be generated by calling the EventPred function. The calculations will be performed using 1,000 simulation runs. A list of all required parameters (parameters) needs to be created as shown below and then passed to this function. A simulation report can be generated using the GenerateReport function and a graphical user interface can be launched using the EventPredApp function.

Arguments

No arguments

Value

No return value

See Also

EventPred

Examples

```
# List of all parameters
parameters = list()

# Load the built-in data set with the patient enrollment, event and dropout
# information (EventPredData)

parameters$data_set = EventPredData

# Future time points for computing event predictions
parameters$time_points = seq(from = 12, to = 24, by = 1)

# Prior distribution for the event hazard rate based on the
# expected median time of 15 months and the uncertainty parameter of 0.3
parameters$event_prior_distribution =
    EventPredPriorDistribution(expected = log(2) / 15, uncertainty = 0.3)

# Prior distribution for the patient dropout hazard rate based on the
# expected median time of 80 months and the uncertainty parameter of 0.3
parameters$dropout_prior_distribution =
    EventPredPriorDistribution(expected = log(2) / 80, uncertainty = 0.3)

# Prior distribution for the patient enrollment rate based on the
# expected enrollment rate of 35 patients per month and the uncertainty
# parameter of 0.3
parameters$enrollment_prior_distribution =
    EventPredPriorDistribution(expected = 35, uncertainty = 0.3)

# Number of simulations, you should prefer more
parameters$nsims = 100

# Remove this parameter in your code:
parameters$withoutCharts = TRUE

# Forecast the number of events at the pre-defined time points
results = EventPred(parameters)

# Generate a simulation report (remove tempfile)
GenerateReport(results,
               tempfile("EventPred.docx", fileext=".docx"))
```

EventPredPriorDistribution

Calculation of the parameters of prior gamma distributions

Description

This function can be used for calculating the parameters of prior gamma distributions for the event hazard rate, patient dropout hazard rate and patient enrollment rate in the EventPred function. For an example of the function call, see EventPredExample.

Usage

```
EventPredPriorDistribution(expected, uncertainty)
```

Arguments

expected	Numeric value defining the expected value of the event hazard rate, patient dropout hazard rate or patient enrollment rate. This value must be positive.
uncertainty	Numeric value defining the amount of uncertainly around the expected value defined above. As a general guideline, the uncertainty parameters of 0.1, 0.3 and 0.5 define high-confidence, medium-confidence and low-confidence prior distributions. This value must be between 0 and 1.

Value

The function returns a numeric vector with the shape and rate parameters of the prior gamma distribution.

See Also

EventPred

FutRule *Simulation-based selection of an optimal futility stopping rule at an interim analysis*

Description

This function evaluates operating characteristics of a multi-arm trial design with a single interim analysis. A futility stopping rule will be applied at this interim look and the trial will be stopped early for futility if the predicted probability of success (conditional power) is less than a pre-defined futility threshold in all treatment arms. An optimal value of the futility threshold is derived by maximizing the sensitivity and specificity rates. For examples of the function call, see FutRuleExample1, FutRuleExample2 or FutRuleExample3.

Usage

```
FutRule(parameters)
```

Arguments

parameters	List of the trial design and other parameters. The required elements are defined below: • endpoint_type: Character value defining the primary endpoint's type. Possible values: – "Normal": Normally distributed endpoint. – "Binary": Binary endpoint.

  - "Time-to-event": Time-to-event endpoint.
- direction: Character value defining the direction of favorable outcome. Possible values: "Higher" (a higher value of the endpoint indicates a more favorable outcome) and "Lower" (a lower value of the endpoint indicates a more favorable outcome).
- sample_size: Integer vector defining the number of enrolled patients in the trial arms (control and multiple experimental treatments). Each element must be positive.
- control_mean: Numeric value defining the mean of the primary endpoint in the control arm. This parameter is required only with normally distributed endpoints (endpoint_type="Normal").
- control_sd: Numeric value defining the standard deviation of the primary endpoint in the control arm. This value must be positive. This parameter is required only with normally distributed endpoints.
- treatment_mean: Numeric vector defining the means of the primary endpoint in the experimental treatment arms. This parameter is required only with normally distributed endpoints.
- treatment_sd: Numeric vector defining the standard deviations of the primary endpoint in the experimental treatment arms. Each element must be positive. This parameter is required only with normally distributed endpoints.
- control_rate: Numeric value defining the proportion or response rate for the primary endpoint in the control arm. This value must be between 0 and 1. This parameter is required only with binary endpoints (endpoint_type="Binary").
- treatment_rate: Numeric vector defining the proportions or response rates for the primary endpoint in the experimental treatment arms. Each element must be between 0 and 1. This parameter is required only with binary endpoints.
- control_time: Numeric value defining the median time, e.g., the median survival time, for the exponentially distributed primary endpoint in the control arm. This value must be positive. This parameter is required only with time-to-event endpoints (endpoint_type="Time-to-event").
- treatment_time: Numeric vector defining the median times, e.g., the median survival times, for the exponentially distributed primary endpoint in the experimental treatment arms. Each element must be positive. This parameter is required only with time-to-event endpoints.
- info_frac: Numeric value defining the information fraction at the interim analysis. This value must be between 0 and 1.
- event_count: Numeric value defining the total number of events at the final analysis. This value must be positive. This parameter is required only with time-to-event endpoints.
- dropout_rate: Numeric value defining the patient dropout rate. With normally distributed endpoints and binary endpoints, a uniform patient dropout process is assumed and thus this parameter defines the fraction of patients that will be excluded from the interim and final analyses. With time-to-event endpoints, the time to patient dropout is assumed to follow an expo-

nential distribution and this parameter defines the annual dropout rate. This value must be between 0 and 1.

- `enrollment_period`: Numeric value defining the length of the patient enrollment period. This value must be positive. This parameter is required only with time-to-event endpoints.
- `enrollment_parameter`: Numeric value defining the median enrollment time. The patient enrollment process is assumed to be governed by a truncated exponential distribution and this parameter defines the time point by which 50% of the patients are enrolled into the trial. This value must be between 0 and the length of the patient enrollment period. This parameter is required only with time-to-event endpoints..
- `alpha`: Numeric value defining the overall one-sided Type I error rate. The default value is 0.025.
- `random_seed`: Integer value defining the random number generator seed. The default value is 49283.
- `nsims`: Integer value defining the number of simulation runs.
- `ncores`: Integer value defining the number of cores for parallel calculations. The number of cores cannot exceed the maximum available number of cores. The default value is 1.

Value

The function returns an object of class '`FutRuleResults`'. This object is a list with the following components:

`parameters`	List containing the user-specified parameters.
`sim_summary`	List containing the operating characteristics of the futility stopping rule.

A detailed summary of the simulation results can be created using the `GenerateReport` function.

See Also

`FutRuleApp`

`FutRuleApp` *Graphical user interface for an optimal selection of a futility stopping rule at an interim analysis*

Description

This function creates a web application with a Shiny-based graphical user interface to facilitate the process of defining an optimal futility stopping rule at an interim analysis in a Phase II or Phase III clinical trial. The application supports the same functionality as the `FutRule` function. For an example of the function call, see FutRuleExample1.

Usage

```
FutRuleApp()
```

Arguments

No arguments

Value

No return value, called for side effects

See Also

FutRuleFutRule

`FutRuleExample1` *Simulation-based selection of an optimal futility stopping rule (normally distributed endpoint)*

Description

Consider a multi-arm Phase II or Phase III trial with a single interim analysis and assume that the primary efficacy endpoint is normally distributed and a larger value indicates to a more favorable outcome. A futility assessment will be performed for each treatment arm at this interim look using the predicted probability of success (conditional power). A treatment arm will be dropped at this decision point if conditional power is below a pre-defined threshold. An optimal value of this futility threshold can be found by computing the sensitivity and specificity rates associated with the futility stopping rule and then identifying the threshold that simultaneously maximizes both rates.

The following design parameters will be assumed in the trial:

- Two treatments (two doses of an experimental treatment) will be compared to a control in this trial and a balanced design with 100 enrolled patients per trial arm will be utilized.
- The patient dropout rate at the end of the treatment period is assumed to be 5%.
- The information fraction at the interim analysis is 50%.

The calculations will be performed under the following set of treatment effect assumptions:

- The mean and standard deviation of the primary efficacy endpoint in the control arm are set to 0 and 1, respectively.
- The mean values in the two treatment arms are assumed to be equal to 0.25 and 0.3 with a common standard deviation of 1.

Operating characteristics of the futility stopping rule such as the sensitivity and specificity rates as well as an optimal futility threshold will be computed using the `FutRule` function based on 1,000 simulation runs. A list of all trial design parameters (`parameters`) needs to be set up as shown below and passed to this function. A detailed simulation report can be generated using the `GenerateReport` function and a graphical user interface can be launched by calling the `FutRuleApp` function.

Arguments

No arguments

Value

No return value

See Also

FutRule, FutRuleExample2, FutRuleExample3

Examples

```
# List of all parameters
parameters = list()

# Endpoint type
parameters$endpoint_type = "Normal"

# Number of enrolled patients (control, two treatments)
parameters$sample_size = c(100, 100, 100)

# Direction of favorable outcome
parameters$direction = "Higher"

# Patient dropout rate
parameters$dropout_rate = 0.05

# Mean and SD in the control arm
parameters$control_mean = 0
parameters$control_sd = 1

# Means and SDs in the treatment arms
parameters$treatment_mean = c(0.25, 0.30)
parameters$treatment_sd = c(1, 1)

# Information fraction
parameters$info_frac = 0.5

# One-sided alpha level
parameters$alpha = 0.025

# Number of simulations, you should prefer more
parameters$nsims = 100

# Remove this parameter in your code:
parameters$withoutCharts = TRUE

# Run simulations to compute characteristics of the futility stopping rule
results = FutRule(parameters)

# Generate a simulation report (remove tempfile)
GenerateReport(results,
               tempfile("FutRule Normally distributed endpoint.docx", fileext=".docx"))
```

FutRuleExample2 *Simulation-based selection of an optimal futility stopping rule (binary endpoint)*

Description

Consider a Phase II or Phase III trial with three treatment arms that correspond to two doses of an experimental treatment versus a control and assume that the primary efficacy endpoint is a binary endpoint (response rate, i.e., a higher value indicates a more favorable outcome). A futility assessment will be performed for each treatment arm at an interim look using conditional power and a treatment arm will be dropped if conditional power is below a pre-defined futility threshold. An optimal value of the futility threshold can be found by computing the sensitivity and specificity rates associated with the futility stopping rule and then identifying the threshold that simultaneously maximizes both rates.

The following design parameters will be assumed in the trial:

- A balanced design with 75 enrolled patients per trial arm will be utilized.
- The patient dropout rate at the end of the treatment period is assumed to be 5%.
- An early interim analysis with the information fraction of 30% will be conducted.

The calculations will be performed under the following set of treatment effect assumptions:

- The control response rate is set to 35%.
- The response rates in the three treatment arms are set to 45%, 50% and 55%.

Operating characteristics of the futility stopping rule (sensitivity and specificity rates) and an optimal futility threshold will be computed using the FutRule function based on 1,000 simulation runs. A list of all trial design parameters (parameters) needs to be set up as shown below and passed to this function. A detailed simulation report can be generated using the GenerateReport function and a graphical user interface can be launched by calling the FutRuleApp function.

Arguments

No arguments

Value

No return value

See Also

FutRule, FutRuleExample1, FutRuleExample3

Examples

```
# List of all parameters
parameters = list()

# Endpoint type
parameters$endpoint_type = "Binary"

# Direction of favorable outcome
parameters$direction = "Higher"

# Number of enrolled patients (control, three treatments)
parameters$sample_size = c(75, 75, 75, 75)

# Dropout rate
parameters$dropout_rate = 0.05

# Response rate in the control arm
parameters$control_rate = 0.35

# Response rates in the treatment arms
parameters$treatment_rate = c(0.45, 0.5, 0.55)

# Information fraction
parameters$info_frac = 0.3

# One-sided alpha level
parameters$alpha = 0.025

# Number of simulations, you should prefer more
parameters$nsims = 100

# Remove this parameter in your code:
parameters$withoutCharts = TRUE

# Run simulations to compute characteristics of the futility stopping rule
results = FutRule(parameters)

# Generate a simulation report (remove tempfile)
GenerateReport(results,
               tempfile("FutRule Binary endpoint.docx", fileext=".docx"))
```

FutRuleExample3 *Simulation-based selection of an optimal futility stopping rule (time-to-event endpoint)*

Description

Consider a Phase III trial with a time-to-event primary efficacy endpoint. A single regimen of an experimental treatment will be evaluated versus a control. A futility assessment will be conducted

at an interim look using conditional power and the trial will be stopped for futility if conditional power in the single treatment arm is below a pre-defined futility threshold. An optimal value of the futility threshold can be found by computing the sensitivity and specificity rates associated with the futility stopping rule and then identifying the threshold that simultaneously maximizes both rates.

The following design parameters will be assumed in the trial:

- A balanced design with 125 enrolled patients per trial arm will be used.
- The target number of events at the final analysis is 175.
- The annual patient dropout rate is equal to 5% (the time to patient dropout is assumed to follow an exponential distribution).
- The length of the patient enrollment period is 12 months and the median enrollment time is expected to be 9 months.
- A late interim analysis with the information fraction of 60% will be conducted.

The calculations will be performed assuming the median event times of 7.5 months and 10.5 months in the control and treatment arms, respectively.

Operating characteristics of the futility stopping rule (sensitivity and specificity rates) and an optimal futility threshold will be computed using the `FutRule` function based on 1,000 simulation runs. A list of all trial design parameters (`parameters`) needs to be set up as shown below and passed to this function. A detailed simulation report can be generated using the `GenerateReport` function and a graphical user interface can be launched by calling the `FutRuleApp` function.

Arguments

No arguments

Value

No return value

See Also

`FutRule`, `FutRuleExample1`, `FutRuleExample2`

Examples

```
# List of all parameters
parameters = list()

# Endpoint type
parameters$endpoint_type = "Time-to-event"

# Direction of favorable outcome
parameters$direction = "Higher"

# Number of enrolled patients (control, treatment)
parameters$sample_size = c(125, 125)

# Target event count at the final analysis
```

```
parameters$event_count = 175

# Annual patient dropout rate
parameters$dropout_rate = 0.05

# Median time in the control arm
parameters$control_time = 7.5

# Median time in the treatment arm
parameters$treatment_time = 10.5

# Information fraction
parameters$info_frac = 0.6

# Enrollment period
parameters$enrollment_period = 12

# Median enrollment time
parameters$enrollment_parameter = 9

# One-sided alpha level
parameters$alpha = 0.025

# Number of simulations, you should prefer more
parameters$nsims = 100

# Remove this parameter in your code:
parameters$withoutCharts = TRUE

# Run simulations to compute characteristics of the futility stopping rule
results = FutRule(parameters)

# Generate a simulation report (remove tempfile)
GenerateReport(results,
               tempfile("FutRule Time-to-event endpoint.docx", fileext=".docx"))
```

GenerateReport *Simulation report*

Description

This function creates a detailed simulation report in a Microsoft Word format. For examples of the function call, see ADSSModExample1, ADTreatSelExample1, ADPopSelExample1, FutRuleExample1, EventPredExample and ADRandExample.

Usage

```
GenerateReport(results, report_filename)
```

Arguments

`results`	Object created by a simulation function, e.g., `ADSSMod`.
`report_filename`	Character value defining the report's filename. The report is saved in the current working directory.

Value

No return value, called for side effects

`MultAdj` *Simulation-based design of traditional trials with multiple objectives*

Description

This function supports power calculations for a broad class of Phase III clinical trials with multiple objectives, including

- Trials with a single source of multiplicity, i.e., trials with a single endpoint and several dose-control comparisons, or two-arm trials with several endpoints.
- Trials with several sources of multiplicity, i.e., trials with several endpoints and several dose-placebo comparisons.

A fixed-sample design is assumed in this setting. Most commonly used multiplicity adjustments are supported for trials with a single source of multiplicity. In addition, global testing procedures can be applied in two-arm trials with several endpoints. Popular gatekeeping procedures are supported in the advanced multiplicity problems with several sources. For examples of the function call, see MultAdjExample1, MultAdjExample2 or MultAdjExample3.

Usage

```
MultAdj(parameters)
```

Arguments

`parameters`	List of the trial design and other parameters. The required elements are defined below: • `endpoint_type`: Character value defining the common type of trial endpoints. Possible values: – `"Normal"`: Normally distributed endpoint. – `"Binary"`: Binary endpoint. • `direction`: Character value defining the common direction of favorable outcome for all endpoints. Possible values: `"Higher"` (a higher value of each endpoint indicates a more favorable outcome) and `"Lower"` (a lower value of each endpoint indicates a more favorable outcome).

- `n_comparisons`: Integer value defining the number of dose-control comparisons in the trial. This value must be positive.
- `n_endpoints`: Integer value defining the number of endpoints in the trial. This value must be positive. Either `n_comparisons` or `n_endpoints` must be greater than 1.
- `sample_size`: Integer vector defining the number of enrolled patients in each trial arm (control and experimental treatments). Each element must be positive.
- `control_mean`: Numeric vector defining the mean of each endpoint in the control arm. This parameter is required only with normally distributed endpoints (`endpoint_type="Normal"`).
- `control_sd`: Numeric vector defining the standard deviation of each endpoint in the control arm. Each element must be positive. This parameter is required only with normally distributed endpoints.
- `treatment_mean`: Numeric vector or matrix defining the mean of each endpoint in each experimental treatment arm. In clinical trials with several endpoints and several dose-placebo comparisons, the rows corresponds to the endpoints and the columns corresponds to the treatment-control comparisons. This parameter is required only with normally distributed endpoints.
- `treatment_sd`: Numeric vector or matrix defining the standard deviation of each endpoint in each experimental treatment arm. In clinical trials with several endpoints and several dose-placebo comparisons, the rows corresponds to the endpoints and the columns corresponds to the treatment-control comparisons. Each element must be positive. This parameter is required only with normally distributed endpoints.
- `control_rate`: Numeric vector defining the proportion or response rate for each endpoint in the control arm. Each element must be between 0 and 1. This parameter is required only with binary endpoints (`endpoint_type="Binary"`).
- `treatment_rate`: Numeric vector or matrix defining the proportion or response rate for each endpoint in each experimental treatment arm. In clinical trials with several endpoints and several dose-placebo comparisons, the rows corresponds to the endpoints and the columns corresponds to the treatment-control comparisons. Each element must be between 0 and 1. This parameter is required only with binary endpoints.
- `endpoint_correlation`: Numeric matrix defining the pairwise correlations among the endpoint-specific test statistics. Each element must be between 0 and 1 and the matrix must be positive definite. This parameter is required only in trials with multiple endpoints.
- `mult_test`: Character value defining the multiple testing procedure, global testing procedure or gatekeeping procedure. Possible values:
  - `"Bonferroni"`: Bonferroni multiple testing procedure.
  - `"Holm"`: Holm multiple testing procedure in trials with a single source of multiplicity or Holm-based gatekeeping procedure in trials with several sources of multiplicity.
  - `"Fixed-sequence"`: Fixed-sequence multiple testing procedure.
  - `"Chain"`: Chain multiple testing procedure.

  - "Hochberg": Hochberg multiple testing procedure in trials with a single source of multiplicity or Hochberg-based gatekeeping procedure in trials with several sources of multiplicity.
  - "Hommel": Hommel multiple testing procedure in trials with a single source of multiplicity or Hommel-based gatekeeping procedure in trials with several sources of multiplicity.
  - "O'Brien": O'Brien global testing procedure.

  Note that the O'Brien procedure can be used only in two-arm trials with several endpoints, similarly gatekeeping procedures can be used only in trials with several endpoints and several dose-placebo comparisons.

- weights: Numeric vector defining the initial hypothesis weights. Each element must be between 0 and 1. This parameter is required only with multiple testing procedures.
- transition: Numeric matrix defining the hypothesis transition parameters. Each element must be between 0 and 1 and the sum of elements in each row must be less than or equal to 1. This parameter is required only with the chain multiple testing procedure.
- sequence: Integer vector defining the hypothesis testing sequence. This parameter is required only with the fixed-sequence multiple testing procedures.
- mult_method: Character value defining the mixture method for the gatekeeping procedure. Possible values:
  - "Standard": Standard mixture method.
  - "Modified": Modified mixture method.
  - "Enhanced": Enhanced mixture method.

  This parameter is required only with gatekeeping procedures.
- mult_test_gamma: Numeric vector defining the truncation parameter for each endpoint-specific family of hypotheses. The vector's length must be equal to the number of endpoints. Each element must be between 0 and 1, the last element may be equal to 1 whereas the other elements must be strictly less than 1. This parameter is required only with gatekeeping procedures.
- dropout_rate: Numeric value defining the patient dropout rate. A uniform patient dropout process is assumed and thus this parameter defines the fraction of patients that will be excluded from the analysis. This value must be between 0 and 1.
- alpha: Numeric value defining the overall one-sided Type I error rate. The default value is 0.025.
- random_seed: Integer value defining the random number generator seed. The default value is 49283.
- nsims: Integer value defining the number of simulation runs.
- ncores: Integer value defining the number of cores for parallel calculations. The number of cores cannot exceed the maximum available number of cores. The default value is 1.

Value

The function returns an object of class 'MultAdjResults'. This object is a list with the following components:

parameters	List containing the user-specified parameters.
sim_results	Data frame containing the raw and adjusted p-values generated by the hypothesis tests for each simulation run. The first set of n columns correspond to the raw p-values for the n null hypotheses and the next set of n columns correspond to the adjusted p-values for the n null hypotheses.
sim_summary	List containing the power calculation results for the specified multiple testing procedure, global testing procedure or gatekeeping procedure.

A detailed summary of the simulation results can be created using the GenerateReport function.

See Also

MultAdjApp, MultAdjExample1, MultAdjExample2, MultAdjExample3

MultAdjApp *Graphical user interface for power calculations in traditional trials with multiple objectives*

Description

This function creates a web application with a Shiny-based graphical user interface to perform power calculations in traditional trials with multiple objectives. The application supports the same functionality as the MultAdj function. For an example of the function call, see MultAdjExample1.

Usage

```
MultAdjApp()
```

Arguments

No arguments

Value

No return value, called for side effects

See Also

MultAdj, MultAdj, MultAdjExample2, MultAdjExample3

`MultAdjExample1` *Simulation-based power calculations in Phase III trials with multiple dose-placebo comparisons*

Description

Consider a Phase III trial in patients with rheumatoid arthritis that will be conducted to investigate the efficacy and safety of three doses of an experimental treatment versus placebo. The three doses will be labeled as the low, medium and high doses. The primary efficacy endpoint is a binary endpoint based on the American College of Rheumatology response criteria (ACR50). The endpoint represents the proportion of patients who experience a symptomatic improvement after 26 weeks of treatment and thus a higher value of the endpoint indicates a beneficial effect.

A multiplicity adjustment will be applied to address Type I error rate inflation cased by the evaluation of three dose-placebo comparisons. The adjustment will rely on the fixed-sequence procedure with the following sequence of dose-placebo comparisons:

- High dose versus placebo.
- Medium dose versus placebo.
- Low dose versus placebo.

Each comparison in this sequence will be evaluated at a one-sided alpha of 0.025 (provided the preceding comparison was significant) to control the overall Type I error rate at a one-sided 2.5% level.

The following design parameters will be used in this example:

- A total of 440 patients will be enrolled in the trial and randomly assigned to placebo and three doses of the experimental treatment using an equal randomization approach.
- The patient dropout rate at the end of the 26-week treatment period is expected be 20%.

Power calculations will be performed in this trial using the `MultAdj` function. A list of all design and multiplicity adjustment parameters (`parameters`) needs to be created and then passed to this function. A simulation report can be created by calling the `GenerateReport` function. Power calculations can also be performed in this trial using a Shiny-based application with a graphical user interface that can be launched by calling the `MultAdjApp` function.

Arguments

No arguments

Value

No return value

See Also

`MultAdjApp`, `MultAdj`, `MultAdjExample2`, `MultAdjExample3`

Examples

```
# List of all parameters
parameters = list()

# Endpoint type
parameters$endpoint_type = "Binary"

# Direction of beneficial effect
parameters$direction = "Higher"

# Number of dose-placebo comparisons
parameters$n_comparisons = 3

# Number of endpoints
parameters$n_endpoints = 1

# Number of enrolled patients (placebo, low dose, medium dose and high dose)
parameters$sample_size = c(110, 110, 110, 110)

# Patient dropout rate
parameters$dropout_rate = 0.2

# Endpoint information
parameters$control_rate = 0.3
parameters$treatment_rate = c(0.5, 0.55, 0.6)

# Multiple testing procedure
parameters$mult_test = "Fixed-sequence"

# Hypothesis testing sequence (the null hypotheses of no effect will be
# tested in the following sequence: H3 [high dose versus placebo], H2 [medium dose
# versus placebo] and H1 [low dose versus placebo])
parameters$sequence = c(3, 2, 1)

# Overall one-sided Type I error rate
parameters$alpha = 0.025

# Number of simulations
parameters$nsims = 1000

# Run simulations to perform power calculations
results = MultAdj(parameters)

# Generate a simulation report (remove tempfile)
GenerateReport(results,
               tempfile("MultAdj Multiple dose-placebo comparisons.docx",
               fileext=".docx"))
```

`MultAdjExample2` *Simulation-based power calculations in Phase III trials with multiple endpoints*

Description

Consider a Phase III trial for the treatment of agitation in patients with dementia. A single dose of an experimental treatment will be compared to placebo in this trial. The efficacy profile of the experimental treatment will be assessed using three primary endpoints, namely, three components of the Cohen-Mansfield Agitation Inventory, Community version (CMAI-C) rating scale:

- Aggressive behavior.
- Non-aggressive agitated behavior.
- Verbally agitated behavior.

Changes in the individual endpoints will be assessed at Day 28. Each endpoint is normally distributed and, since higher scores on this rating scale indicate more severe symptoms, a lower value of each endpoint corresponds to a beneficial effect. The endpoints are assumed to be equally correlated with a common correlation coefficient of 0.4.

An important feature of this trial is that the trial will meet its primary objective if at least one endpoint test is significant. As a result, the overall Type I error rate will be inflated if the primary analysis is performed without a multiplicity adjustment. To preserve the overall Type I error rate at a one-sided 2.5% level, the Holm procedure will be applied. The three endpoints are considered to be equally important and, by default, equal weights will be assigned to the endpoint-specific null hypotheses of no treatment effect.

The following design parameters will be used in this example:

- A total of 380 patients will be enrolled in the trial to be assigned to either placebo or experimental treatment using a 1:1 ratio.
- The patient dropout rate at the end of the treatment period is expected be 5%.

The `MultAdj` function will be invoked to run power calculations in this trial. A list of all design and multiplicity adjustment parameters (`parameters`) will be created and passed to this function. A simulation report will be created by calling the `GenerateReport` function. Power calculations can also be performed in this trial using a Shiny-based application that can be launched by calling the `MultAdjApp` function.

Arguments

No arguments

Value

No return value

See Also

`MultAdjApp`, `MultAdj`, `MultAdjExample1`, `MultAdjExample3`

Examples

```
# List of all parameters
parameters = list()

# Endpoint type
parameters$endpoint_type = "Normal"

# Direction of beneficial effect
parameters$direction = "Lower"

# Number of dose-placebo comparisons
parameters$n_comparisons = 1

# Number of endpoints
parameters$n_endpoints = 3

# Number of enrolled patients (placebo, experimental treatment)
parameters$sample_size = c(190, 190)

# Patient dropout rate
parameters$dropout_rate = 0.05

parameters$control_mean = c(0, 0, 0)
parameters$treatment_mean = c(-0.3, -0.3, -0.3)
parameters$control_sd = c(1, 1, 1)
parameters$treatment_sd = c(1, 1, 1)

# Endpoint correlation matrix
parameters$endpoint_correlation = matrix(c(1, 0.4, 0.4,
                                           0.4, 1, 0.4,
                                           0.4, 0.4, 1), 3, 3)

# Multiple testing procedure
parameters$mult_test = "Holm"

# Overall one-sided Type I error rate
parameters$alpha = 0.025

# Number of simulations
parameters$nsims = 1000

# Run simulations to perform power calculations
results = MultAdj(parameters)

# Generate a simulation report (remove tempfile)
GenerateReport(results,
               tempfile("MultAdj Multiple endpoints.docx",
               fileext=".docx"))
```

`MultAdjExample3` *Simulation-based power calculations in Phase III trials with multiple endpoints and multiple dose-placebo comparisons*

Description

The advanced setting with several sources of multiplicity, i.e., multiplicity induced by the analysis of multiple endpoints as well as multiple dose-placebo comparisons, will be illustrated using a Phase III trial for the treatment of acute schizophrenia. The efficacy profiles of two doses of an experimental treatment (Dose L and Dose H) will be compared to that of placebo using the following two endpoints:

- Primary endpoint: Change in the Positive and Negative Syndrome Scale (PANSS) total score.
- Key secondary endpoint: Change in the Clinical Global Impressions severity (CGI-S) score.

The efficacy evaluations will be performed at the end of the 4-week treatment period. Both endpoints are normally distributed and a lower value of each endpoint indicates a more favorable outcome. The correlation between the endpoint-specific test statistics is assumed to be 0.5.

The resulting four null hypotheses of no effect will be grouped into two endpoint-specific families:

- Family 1: H1 (No difference between Dose L and placebo on the primary endpoint) and H2 (No difference between Dose H and placebo on the primary endpoint).
- Family 2: H3 (No difference between Dose L and placebo on the key secondary endpoint) and H4 (No difference between Dose H and placebo on the key secondary endpoint).

Clinically relevant logical restrictions will be imposed on the testing strategy to ensure that H3 can be tested only if H1 is rejected and H4 can be tested only if H2 is rejected.

The Hochberg-based gatekeeping procedure will be utilized in this trial to control the overall Type I error rate at a one-sided 2.5% level across the four hypotheses of interest. This gatekeeping procedure will be set up using Hochberg-based component procedures, i.e., a truncated version of the Hochberg procedure with the truncation parameter of 0.8 will be applied in Family 1 and the regular Hochberg procedure in Family 2. The modified mixture method will be used in this gatekeeping procedure.

The following design parameters will be used in this example:

- A total of 360 patients will be enrolled in the trial. The patients will be allocated to the three trial arms using a 1:1:1 randomization scheme.
- The patient dropout rate at the end of the 4-week treatment period is expected be 10%.

To compute power for the individual hypotheses as well as each of the two families, the design parameters as well as the parameters of the selected gatekeeping procedure (`parameters`) will passed to the `MultAdj` function. A simulation report will be created by calling the `GenerateReport` function. Power calculations can also be performed in this trial using a Shiny-based application, which can be launched by calling the `MultAdjApp` function.

Arguments

No arguments

Value

No return value

See Also

MultAdjApp, MultAdj, MultAdjExample2, MultAdjExample3

Examples

```
# List of all parameters
parameters = list()

# Endpoint type
parameters$endpoint_type = "Normal"

# Direction of beneficial effect
parameters$direction = "Lower"

# Number of dose-placebo comparisons
parameters$n_comparisons = 2

# Number of endpoints
parameters$n_endpoints = 2

# Number of enrolled patients (placebo, Dose L and Dose H)
parameters$sample_size = c(120, 120, 120)

# Patient dropout rate
parameters$dropout_rate = 0.10

# Endpoint information (rows corresponds to endpoints and
# columns corresponds to dose-placebo comparisons)
parameters$control_mean = c(0, 0)
parameters$treatment_mean = matrix(c(-0.30, -0.35,
                                     -0.35, -0.40), 2, 2, byrow = TRUE)

parameters$control_sd = c(1, 1)
parameters$treatment_sd = matrix(c(1, 1,
                                   1, 1), 2, 2, byrow = TRUE)

# Endpoint correlation matrix
parameters$endpoint_correlation = matrix(c(1, 0.5,
                                           0.5, 1), 2, 2)

# Component procedure to be used in the gatekeeping procedure
parameters$mult_test = "Hochberg"

# Mixture method used in the gatekeeping procedure
parameters$mult_method = "Modified"

# Truncation parameters in the gatekeeping procedure
parameters$mult_test_gamma = c(0.8, 1)
```

```
# Overall one-sided Type I error rate
parameters$alpha = 0.025

# Number of simulations
parameters$nsims = 1000

# Run simulations to perform power calculations
results = MultAdj(parameters)

# Generate a simulation report (remove tempfile)
GenerateReport(results,
               tempfile("MultAdj Multiple endpoints and dose-placebo comparisons.docx",
               fileext=".docx"))
```

Index