Package 'AdaptiveDesignOptimizer'

January 3, 2018

Type Package

Title Optimizes Adaptive Enrichment Designs

Version 0.1.0

Maintainer Michael Rosenblum <mrosen@jhu.edu>

Description This software is intended for investigators planning a confirmatory trial
where it's suspected that a subpopulation may benefit more than the overall population.
The subpopulation could be defined by a risk score or biomarker measured at baseline.
The subpopulation must be defined in advance, e.g., based on prior data or medical knowledge.
Adaptive enrichment designs have potential to provide stronger evidence than
standard designs about treatment benefits for the subpopulation,
its complement, and the combined population.

Imports mvtnorm, plyr, stats, knitr, ggplot2, gridExtra, xtable

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

URL https://github.com/mrosenblum/AdaptiveDesignOptimizer

BugReports https://github.com/mrosenblum/AdaptiveDesignOptimizer/issues

Suggests rmarkdown

Author Michael Rosenblum [cre],
Josh Betz [aut],
Tianchen Qian [aut]

R topics documented:

`optimize_designs` *Adaptive Enrichment Design Optimization Using Simulated Annealing Authors: Josh Betz (jbetz@jhu.edu), Tianchen Qian, Michael Rosenblum*

Description

Adaptive Enrichment Design Optimization Using Simulated Annealing Authors: Josh Betz (jbetz@jhu.edu), Tianchen Qian, Michael Rosenblum

Usage

```
optimize_designs(ui.n.arms, ui.type.of.outcome.data,
  ui.time.to.event.trial.type, ui.time.to.event.non.inferiority.trial.margin,
  ui.subpopulation.1.size, ui.total.alpha, ui.max.size, ui.max.duration,
  ui.accrual.yearly.rate, ui.followup.length, ui.optimization.target,
  ui.time.to.event.censoring.rate, ui.mcid, ui.incorporate.precision.gain,
  ui.relative.efficiency, ui.max.stages, ui.include.designs.start.subpop.1,
  ui.population.parameters, ui.desired.power, ui.scenario.weights,
  min.n.per.arm = 25, min.enrollment.period = 0.5,
  simulated.annealing.parameter.function.scale = 1,
  simulated.annealing.parameter.n.scale = 100,
  simulated.annealing.parameter.period.scale = 2,
  simulated.annealing.parameter.max.iterations = 1000,
  simulated.annealing.parameter.n.simulations = 10000,
  simulated.annealing.parameter.means.temperature = 100,
  simulated.annealing.parameter.survival.temperature = 10,
  simulated.annealing.parameter.evals.per.temp = 10,
  simulated.annealing.parameter.report.iteration = 1,
  simulated.annealing.parameter.power.penalty = 1e+05)
```

Arguments

`ui.n.arms` (the prefix 'ui' abbreviates 'user-input') Number of Arms (including control arm), e.g., 2 arms means one treatment arm and one control arm; the design classes that come with this package can handle 2 or 3 arms

`ui.type.of.outcome.data`
"continuous", "binary", or "time-to-event" (i.e., survival outcome)

`ui.time.to.event.trial.type`
"superiority" or "non-inferiority" trial type; only implemented for time-to-event outcomes

`ui.time.to.event.non.inferiority.trial.margin`
Non-inferiority margin; only relevant if outcome is time-to-event and trial type is non-inferiority. Represented as hazard ratio, required to be at least 1.

`ui.subpopulation.1.size`
Proportion of overall population in subpopulation 1. Must be between 0 and 1.

`ui.total.alpha` Familywise Type I error rate (1-sided)

`ui.max.size` Maximum allowed sample size

`ui.max.duration`
Maximum allowed trial duration in years

`ui.accrual.yearly.rate`
: Number of participants enrolled per year; assumed constant throughout trial

`ui.followup.length`
: Time from enrollment to measurement of primary outcome (only used for continuous or binary outcome types)

`ui.optimization.target`
: Quantity being optimized (objective function); "size" represents expected sample size

`ui.time.to.event.censoring.rate`
: probability that primary outcome is censored, assumed to be independent of the outcome and subpopulation; only implemented for time-to-event outcomes

`ui.mcid`
: Minimum, Clinically Important Treatment Effect (as difference of population means for binary/continuous outcomes; as hazard ratio for time-to-event outcomes)

`ui.incorporate.precision.gain`
: Incorporate into analysis a precision gain from adjustment for prognostic baseline variables; allowed values: TRUE or FALSE

`ui.relative.efficiency`
: If ui.incorporate.precision.gain==TRUE, this specifies relative efficiency (number > 1), representing the assumed precision gain from adjustment for prognostic baseline variables.

`ui.max.stages`
: Maximum number of stages allowed (currently not used by default classes of trial designs)

`ui.include.designs.start.subpop.1`
: Search over designs that allow only subpopulation 1 to be enrolled during stage 1; TRUE or FALSE

`ui.population.parameters`
: Matrix encoding scenarios (data generating distributions) used to define power constraints and objective function

`ui.desired.power`
: Matrix encoding power requirements for each scenario

`ui.scenario.weights`
: Matrix encoding weights used to define objective function

`min.n.per.arm`
: Minimum sample size per arm allowed

`min.enrollment.period`
: Minimum enrollment duration for trial

`simulated.annealing.parameter.function.scale`
: Used by Simulated Annealing Optimization algorithm

`simulated.annealing.parameter.n.scale`
: Used by Simulated Annealing Optimization algorithm

`simulated.annealing.parameter.period.scale`
: Used by Simulated Annealing Optimization algorithm

`simulated.annealing.parameter.max.iterations`
: Maximum Number of Different Designs to Search Over using Simluated Annealing optimization

`simulated.annealing.parameter.n.simulations`
: Used by Simulated Annealing Optimization algorithm

`simulated.annealing.parameter.means.temperature`
: Used by Simulated Annealing Optimization algorithm

`simulated.annealing.parameter.survival.temperature`
: Used by Simulated Annealing Optimization algorithm

`simulated.annealing.parameter.evals.per.temp`
: Used by Simulated Annealing Optimization algorithm

`simulated.annealing.parameter.report.iteration`
: Used by Simulated Annealing Optimization algorithm

`simulated.annealing.parameter.power.penalty`
: Used in Objective Function to incorporate Power Constraints by Simulated Annealing Optimization algorithm

Value

4 element list containing optimized designs from four classes (with increasing complexity):

Designs

(first two not adaptive; last two adaptive) Single.Stage.Equal.Alpha.Allocation.Design

Single.Stage.Optimized.Alpha.Allocation.Design

Two.Stage.Equal.Alpha.Allocation.Design

Two.Stage.Optimized.Alpha.Allocation.Design

Each optimized design is a list containing: design.parameters and design.performance

design.parameters

design.parameters has the following elements: cumulative.sample.sizes.and.calendar.time.per.stage The cumulative number enrolled (if no early stopping) per stage and calendar times of analyses just after each stage. In column names, "A" and "C" denote the treatment arm and control arm, respectively; numbers 1 and 2 indicate the corresponding subpopulation. Sample sizes represent the number enrolled at the time of the corresponding analysis (which may exceed the number of participants with outcomes observed, due to the time between enrollment and outcome measurement for each participant

alpha.allocation=Alpha allocation using Error Spending Approach

futility.boundaries=Boundaries for stopping subpopulation accrual, on the z-scale (or in designs with more than one treatment arm compared to control, this is gives for each treatment arm by subpopulation combination.

design.performance

design.performance contains the following values: Power=Power to reject each null hypothesis under each scenario (NA indicates null hypothesis is true, so no power is presented)

Type.1.Error=Type I error for each null hypothesis under each scenario (NA indicates null hypothesis is false)

Expected.Sample.Size

Expected.Duration (in years)

Distribution.of.sample.size.and.duration.per.scenario=For each scenario, every possible combination of early stopping is considered. Columns C1, A1, etc. have the same meaning as described about for sample.sizes.and.calendar.time.per.stage. The value listed under each such column gives the analysis number at which accrual for that arm by subpopulation combination is stopped. E.g., C1=2,C2=1,A1=2,A2=1 corresponds to stopping the control and treatment A for subpopualtion 1 at the end of stage 1, while these continue to the end of stage 2 for subpopulation 2. The subsequent

columns give the sample size, duration, and person-time when this pattern occurs. The columns frequency and proportion tell how often this pattern occurred under the corresponding scenario number (based on simulation).

Examples

```
#For demonstration purposes, the examples below only execute 2 iterations of simulated annealing.
#In general, it is recommended to use at least 500 iterations.
#Example 1: Time-to-event outcome; 1 treatment arm versus control; non-inferiority design
optimized_designs <- optimize_designs(
  ui.n.arms=2,
  ui.type.of.outcome.data="time-to-event",
  ui.time.to.event.trial.type="non-inferiority",
  ui.time.to.event.non.inferiority.trial.margin=1.35,
  ui.subpopulation.1.size=0.33,
  ui.total.alpha=0.05,
  ui.max.size=10000,
  ui.max.duration=10,
  ui.accrual.yearly.rate=1000,
  ui.followup.length=0,
  ui.optimization.target="size",
  ui.time.to.event.censoring.rate=0,
  ui.mcid=0.1,
  ui.incorporate.precision.gain=FALSE,
  ui.relative.efficiency=1,
  ui.max.stages=2,
  ui.include.designs.start.subpop.1=FALSE,
  ui.population.parameters= 0.08*matrix(c(1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.00, 1.3500001,
    1.00, 1.00, 1.00, 2.14, 1.00, 1.00, 1.3500001, 1.3500001), ncol=4, byrow=TRUE,
    dimnames=list(c(),c("lambda1_con","lambda2_con","lambda1_trt","lambda2_trt"))),
  ui.desired.power=0.8*matrix(c(1.00, 1.00, 0, 1.00, 0, 0, 1.00, 0, 0, 0, 0, 0), ncol=3,
   byrow=TRUE,dimnames=list(c(),c("Pow_H(0,1)","Pow_H(0,2)","Pow_Reject_H0,1_and_H0,2"))),
  ui.scenario.weights=matrix(rep(0.25,4),ncol=1,dimnames=list(c(),c("weight"))),
  simulated.annealing.parameter.max.iterations=2,
  )

  #Example 2: continuous outcome; 1 treatment arm versus control; superiority design
optimized_designs <- optimize_designs(
  ui.n.arms=2,
  ui.type.of.outcome.data="continuous",
  ui.time.to.event.trial.type="",
  ui.time.to.event.non.inferiority.trial.margin=NULL,
  ui.subpopulation.1.size=0.5,
  ui.total.alpha=0.05,
  ui.max.size=1000,
  ui.max.duration=5,
  ui.accrual.yearly.rate=250,
  ui.followup.length=1,
  ui.optimization.target="size",
  ui.time.to.event.censoring.rate=0,
  ui.mcid=NULL,
  ui.incorporate.precision.gain=FALSE,
  ui.relative.efficiency=1,
  ui.max.stages=5,
  ui.include.designs.start.subpop.1=FALSE,
  ui.population.parameters=matrix(c(15,15,3600,3600,3600,3600,15,0,3600,3600,3600,3600,
   0,15,3600,3600,3600,3600,0,0,3600,3600,3600,3600),nrow=4, ncol=6, byrow=TRUE,dimnames=
```

```
    list(c(),c("delta1","delta2","sigma1_trt","sigma1_con","sigma2_trt","sigma2_con"))),
  ui.desired.power=matrix(c(0,0,0.8,0.8,0,0,0,0.8,0,0,0,0), nrow=4, ncol=3, byrow=TRUE,
    dimnames=list(c(),c("Pow_H(0,1)","Pow_H(0,2)","Pow_Reject_H0,1_and_H0,2"))),
  ui.scenario.weights=matrix(c(0.25,0.25,0.25,0.25),ncol=1,dimnames=list(c(),c("weight"))),
  simulated.annealing.parameter.max.iterations=2
)

 #Example 3: binary outcome; 1 treatment arm versus control; superiority design
optimized_designs <- optimize_designs(
  ui.n.arms=2,
  ui.type.of.outcome.data="binary",
  ui.time.to.event.trial.type="",
  ui.time.to.event.non.inferiority.trial.margin=NULL,
  ui.subpopulation.1.size=0.4,
  ui.total.alpha=0.05,
  ui.max.size=2000,
  ui.max.duration=5,
  ui.accrual.yearly.rate=400,
  ui.followup.length=0,
  ui.optimization.target="size",
  ui.time.to.event.censoring.rate=0,
  ui.mcid=NULL,
  ui.incorporate.precision.gain=TRUE,
  ui.relative.efficiency=1.2,
  ui.max.stages=5,
  ui.include.designs.start.subpop.1=FALSE,
  ui.population.parameters=matrix(c(0.4,0.3,0.5,0.4,0.4,0.3,0.4,0.4,0.3,0.3,0.4,0.4),
   nrow=3, ncol=4, byrow=TRUE,dimnames=list(c(),c("p1_trt","p1_con","p2_trt","p2_con"))),
  ui.desired.power=matrix(c(0,0,0.8,0.8,0,0,0,0,0), nrow=3, ncol=3, byrow=TRUE,
    dimnames=list(c(),c("Pow_H(0,1)","Pow_H(0,2)","Pow_Reject_H0,1_and_H0,2"))),
  ui.scenario.weights=matrix(c(0.33,0.33,0.34),ncol=1,dimnames=list(c(),c("weight"))),
  simulated.annealing.parameter.max.iterations=2
)
```

Index