Package 'curtailment'

August 5, 2021

Title Finds Binary Outcome Designs Using Stochastic Curtailment

Version 0.1.1

Maintainer Martin Law <martin.law@mrc-bsu.cam.ac.uk>

Description Finds single- and two-arm designs using stochastic curtailment, as described by Law et al. (2019) <arXiv:1909.03017> and Law et al. (2021) <doi:10.1002/pst.2067> respectively. Designs can be single-stage or multi-stage. Non-stochastic curtailment is possible as a special case. Desired error-rates, maximum sample size and lower and upper anticipated response rates are inputted and suitable designs are returned with operating characteristics. Stopping boundaries and visualisations are also available. The package can find designs using other approaches, for example designs by Simon (1989) <doi:10.1016/0197-2456(89)90015-9> and Mander and Thompson (2010) <doi:10.1016/j.cct.2010.07.008>. Other features: compare and visualise designs using a weighted sum of expected sample sizes under the null and alternative hypotheses and maximum sample size; visualise any binary outcome design.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.1.1

Imports ggplot2, gridExtra, ggthemes, data.table, pkgcond, stats

NeedsCompilation no

Author Martin Law [aut, cre] (<https://orcid.org/0000-0001-9594-348X>)

Repository CRAN

Date/Publication 2021-08-05 07:50:14 UTC

R topics documented:

`drawDiagram` *drawDiagram*

Description

This function produces both a data frame and a diagram of stopping boundaries. The function takes a single argument: the output from the function singlearmDesign. If the supplied argument contains more than one admissible designs, the user is offered a choice of which design to use.

Usage

```
drawDiagram(findDesign.output, print.row = NULL, xmax = NULL, ymax = NULL)
```

Arguments

`findDesign.output`	Output from either the function singlearmDesign or find2stageDesigns
`print.row`	Choose a row number to directly obtain a plot and stopping boundaries for a particular design realisation. Default is NULL.
`xmax, ymax`	Choose values for the upper limits of the x- and y-axes respectively. Helpful for comparing two design realisations. Default is NULL.

Value

The output is a list of two elements. The first, $diagram, is a ggplot2 object showing how the trial should proceed: when to to undertake an interim analysis, that is, when to check if a stopping boundary has been reached (solid colours) and what decision to make at each possible point (continue / go decision / no go decision). The second list element, $bounds.mat, is a data frame containing three columns: the number of participants at which to undertake an interim analysis (m), and the number of responses at which the trial should stop for a go decision (success) or a no go decision (fail).

Author(s)

Martin Law, <martin.law@mrc-bsu.cam.ac.uk>

Examples

```
output <- singlearmDesign(nmin = 30,
 nmax = 30,
 C = 5,
 p0 = 0.1,
 p1 = 0.4,
 power = 0.8,
```

```
alpha = 0.05)
dig <- drawDiagram(output, print.row = 2)
```

---

drawDiagramGeneric *drawDiagramGeneric*

Description

drawDiagramGeneric

Usage

```
drawDiagramGeneric(n, go, nogo, xmax = NULL, ymax = NULL, ylab = NULL)
```

Arguments

n	Maximum sample size
go	Two-column matrix detailing stopping boundaries for a go decision or efficacy stopping, where the first column contains the number of responses and the second column contains the corresponding number of participants so far.
nogo	Two-column matrix detailing stopping boundaries for a no go decision or futility stopping, where the first column contains the number of responses and the second column contains the corresponding number of participants so far.
xmax	Optional. Maximum value for x axis.
ymax	Optional. Maximum value for y axis.
ylab	Optional. Label for y axis.

Value

An object of class ggplot, showing a visualisation of the maximum sample size and inputted stopping boundaries.

Author(s)

Martin Law, <martin.law@mrc-bsu.cam.ac.uk>

Examples

```
go <- cbind(6:8, rep(8,3))
nogo <- cbind(0:5, rep(8,6))
drawDiagramGeneric(n=8, go=go, nogo=nogo, ylab="Number of successes")
```

`find2stageDesigns` *Find two-stage designs*

Description

This function finds two-stage designs for a given set of design parameters, allowing stopping for benefit at the interim (Mander and Thompson's design) or no stopping for benefit at the interim (Simon's design). It returns not only the optimal and minimax design realisations, but all design realisations that could be considered "best" in terms of expected sample size under p=p0 (EssH0), expected sample size under p=p1 (Ess), maximum sample size (n) or any weighted combination of these three optimality criteria.

Usage

```
find2stageDesigns(nmin, nmax, p0, p1, alpha, power, benefit = FALSE)
```

Arguments

`nmin`	Minimum permitted sample size. Should be a multiple of block size or number of stages.
`nmax`	Maximum permitted sample size. Should be a multiple of block size or number of stages.
`p0`	Probability for which to control the type-I error-rate
`p1`	Probability for which to control the power
`alpha`	Significance level
`power`	Required power (1-beta)
`benefit`	Allow the trial to end for a go decision and reject the null hypothesis at the interim analysis (i.e., the design of Mander and Thompson)

Value

A list of class "curtailment_simon" containing two data frames. The first data frame, $input, has a single row and contains all the inputted values. The second data frame, $all.des, contains one row for each design realisation, and contains the details of each design, including sample size, stopping boundaries and operating characteristics. To see a diagram of any obtained design realisation and its corresponding stopping boundaries, simply call the function drawDiagram with this output as the only argument.

Author(s)

Martin Law, <martin.law@mrc-bsu.cam.ac.uk>

References

doi: 10.1016/j.cct.2010.07.008A.P. Mander, S.G. Thompson, Two-stage designs optimal under the alternative hypothesis for phase II cancer clinical trials, Contemporary Clinical Trials, Volume 31, Issue 6, 2010, Pages 572-578

doi: 10.1016/01972456(89)900159Richard Simon, Optimal two-stage designs for phase II clinical trials, Controlled Clinical Trials, Volume 10, Issue 1, 1989, Pages 1-10

Examples

```
find2stageDesigns(nmin=23,
 nmax=27,
 p0=0.75,
 p1=0.92,
 alpha=0.22,
 power=0.95,
 benefit=TRUE)
```

`findLoss` *Find loss scores for a set of single-arm designs*

Description

This function finds loss scores for single-arm designs, in particular, designs found using the function singlearmdesigns(). Weights w0 and w1 are chosen, and the loss for each design is found using the equation *loss = w0*ESS(0) + w1*ESS(1) + (1-w0-w1)*N*

Usage

```
findLoss(main.output, w0, w1)
```

Arguments

`main.output`	Output from function singlearmDesigns().
`w0`	Choice of weight on ESS(0)
`w1`	Choice of weight on ESS(0)

Value

Output is a list identical to the output from function singlearmDesigns(), but with absolute and ranked loss scores added.

Author(s)

Martin Law, <martin.law@mrc-bsu.cam.ac.uk>

Examples

```
output <- singlearmDesign(nmin = 30,
 nmax = 30,
 C = 5,
 p0 = 0.1,
 p1 = 0.4,
 power = 0.8,
 alpha = 0.05)
 output.w.loss <- findLoss(main.output=output,
 w0=0.2,
 w1=0.3)
```

`findNSCdesigns` *findNSCdesigns*

Description

This function finds admissible design realisations for single-arm binary outcome trials, using non-stochastic curtailment. The output is a data frame of admissible design realisations.

Usage

```
findNSCdesigns(nmin, nmax, p0, p1, alpha, power, progressBar = FALSE)
```

Arguments

`nmin`	Minimum permitted sample size.
`nmax`	Maximum permitted sample size.
`p0`	Probability for which to control the type-I error-rate
`p1`	Probability for which to control the power
`alpha`	Significance level
`power`	Required power (1-beta).
`progressBar`	Logical. If TRUE, shows progress bar. Defaults to FALSE.

Value

Output is a list of two dataframes. The first, $input, is a one-row data frame that contains important arguments used in the call. The second, $all.des,contains the operating characteristics of all admissible designs found.

Examples

```
findNSCdesigns(nmin=20, nmax=21, p0=0.1, p1=0.4, alpha=0.1, power=0.8)
```

findSCdesigns *findSCDesigns*

Description

This function finds admissible design realisations for single-arm binary outcome trials, using stochastic curtailment. This function differs from singlearmDesign in that it includes a Simon-style interim analysis after some n1 participants. The output is a data frame of admissible design realisations.

Usage

```
findSCdesigns(
  nmin,
  nmax,
  p0,
  p1,
  alpha,
  power,
  minthetaE = p1,
  maxthetaF = p1,
  bounds = "wald",
  fixed.r1 = NA,
  fixed.r = NA,
  fixed.n1 = NA,
  max.combns = 1e+06,
  maxthetas = NA,
  exact.thetaF = NA,
  exact.thetaE = NA,
  progressBar = FALSE
)
```

Arguments

nmin	Minimum permitted sample size.
nmax	Maximum permitted sample size.
p0	Probability for which to control the type-I error-rate
p1	Probability for which to control the power
alpha	Significance level
power	Required power (1-beta).
minthetaE	Minimum value of upper threshold theta_E_min. Defaults to p.
maxthetaF	Maximum value of lower CP threshold theta_F_max. Defaults to p.
bounds	choose what final rejection boundaries should be searched over: Those of A'Hern ("ahern"), Wald ("wald") or no constraints (NA). Defaults to "wald".
fixed.r1	Choose what interim rejection boundaries should be searched over. Useful for reproducing a particular design realisation. Defaults to NA.

fixed.r	Choose what final rejection boundaries should be searched over. Useful for reproducing a particular design realisation. Defaults to NA.
fixed.n1	Choose what interim sample size values n1 should be searched over. Useful for reproducing a particular design realisation. Defaults to NA.
max.combns	Provide a maximum number of ordered pairs (theta_F, theta_E). Defaults to 1e6.
maxthetas	Provide a maximum number of CP values used to create ordered pairs (theta_F, theta_E). Can be used instead of max.combns. Defaults to NA.
exact.thetaF	Provide an exact value for lower threshold theta_F. Useful for reproducing a particular design realisation. Defaults to NA.
exact.thetaE	Provide an exact value for upper threshold theta_E. Useful for reproducing a particular design realisation. Defaults to NA.
progressBar	Logical. If TRUE, shows progress bar. Defaults to FALSE.

Value

Output is a list of two dataframes. The first, $input, is a one-row data frame that contains important arguments used in the call. The second, $all.des,contains the operating characteristics of all admissible designs found.

Examples

```
findSCdesigns(nmin = 20, nmax = 20, p0 = 0.1, p1 = 0.4, power = 0.8, alpha = 0.1, max.combns=1e2)
```

---

findSingleSimonDesign *Find a single Simon design*

Description

This function finds the operating characteristics of a single realisation of a Simon design. It returns the inputted values along with the type-I error-rate (alpha), power, expected sample size under p=p0 (EssH0) and expected sample size under p=p1 (Ess).

Usage

```
findSingleSimonDesign(n1, n2, r1, r, p0, p1)
```

Arguments

n1	Number of participants in stage 1
n2	Number of participants in stage 2
r1	Interim stopping boundary that must be exceeded to avoid no go stopping
r	Final rejection boundary that must be exceeded to reject the null hypothesis.
p0	Anticipated response probability for inefficacious treatment
p1	Anticipated response probability for efficacious treatment

Value

A vector containing all the inputted values and corresponding operating characteristics.

Author(s)

Martin Law, <martin.law@mrc-bsu.cam.ac.uk>

References

doi: 10.1016/01972456(89)900159Richard Simon, Optimal two-stage designs for phase II clinical trials, Controlled Clinical Trials, Volume 10, Issue 1, 1989, Pages 1-10

Examples

```
findSingleSimonDesign(n1 = 15, n2 = 11, r1 = 1, r = 4, p0 = 0.1, p1 = 0.3)
```

`plotByWeight` *plotByWeight*

Description

This function shows the omni-admissible design – the design realisation with the lowest loss score – from a subset of admissible designs. The input is an object created by singlearmDesign.

Usage

```
plotByWeight(main.output, split.by.cohort.size = TRUE)
```

Arguments

`main.output`	Object created by the function(s) listed above.
`split.by.cohort.size`	Logical. If TRUE, creates separate plots for each cohort size/block size/ number of stages. Defaults to TRUE.

Value

An object of class `ggplot` if either the chosen designs have only one unique cohort size or if split.by.cohort.size is FALSE. Otherwise, a list containing multiple `ggplot` objects is returned. The objects show which design has the lowest loss score, at each possible set of weights (w0, w1). Each design is characterised using the format {r/N, thetaF/thetaE}.

Examples

```
designs <- singlearmDesign(nmin=30, nmax=30, C=5, p0=0.1, p1=0.4, power=0.8, alpha = 0.05)
plotByWeight(designs)
```

`singlearmDesign` *Find single-arm trials using stochastic curtailment*

Description

This function finds admissible design realisations for single-arm binary outcome trials, using stochastic curtailment. The output can be used as the sole argument in the function 'drawDiagram', which will return the stopping boundaries for the admissible design of your choice. Monitoring frequency can set in terms of block(/cohort) size ("C") or number of stages ("stages").

Usage

```
singlearmDesign(
  nmin,
  nmax,
  C = NA,
  stages = NA,
  p0,
  p1,
  alpha,
  power,
  maxthetaF = p1,
  minthetaE = p1,
  bounds = "wald",
  return.only.admissible = TRUE,
  max.combns = 1e+06,
  maxthetas = NA,
  fixed.r = NA,
  exact.thetaF = NA,
  exact.thetaE = NA,
  progressBar = FALSE
)
```

Arguments

`nmin`	Minimum permitted sample size. Should be a multiple of block size or number of stages.
`nmax`	Maximum permitted sample size. Should be a multiple of block size or number of stages.
`C`	Block size. Vectors, i.e., multiple values, are permitted.
`stages`	Number of interim analyses or "stages". Only required if not setting block size C. Vectors, i.e., multiple values, are permitted.
`p0`	Probability for which to control the type-I error-rate
`p1`	Probability for which to control the power
`alpha`	Significance level

power — Required power (1-beta).

maxthetaF — Maximum value of lower CP threshold theta_F_max. Defaults to p1.

minthetaE — Minimum value of upper threshold theta_E_min. Defaults to p1.

bounds — choose what final rejection boundaries should be searched over: Those of A'Hern ("ahern"), Wald ("wald") or no constraints (NA). Defaults to "wald".

return.only.admissible — Logical. Returns only admissible design realisations if TRUE, otherwise returns all feasible designs. Defaults to TRUE.

max.combns — Provide a maximum number of ordered pairs (theta_F, theta_E). Defaults to 1e6.

maxthetas — Provide a maximum number of CP values used to create ordered pairs (theta_F, theta_E). Can be used instead of max.combns. Defaults to NA.

fixed.r — Choose what final rejection boundaries should be searched over. Useful for reproducing a particular design realisation. Defaults to NA.

exact.thetaF — Provide an exact value for lower threshold theta_F. Useful for reproducing a particular design realisation. Defaults to NA.

exact.thetaE — Provide an exact value for upper threshold theta_E. Useful for reproducing a particular design realisation. Defaults to NA.

progressBar — Logical. If TRUE, shows progress bar. Defaults to FALSE.

Value

Output is a list of two dataframes. The first, $input, is a one-row data frame that contains all the arguments used in the call. The second, $all.des, contains the operating characteristics of all admissible designs found.

Author(s)

Martin Law, <martin.law@mrc-bsu.cam.ac.uk>

Examples

```
output <- singlearmDesign(nmin = 30,
 nmax = 30,
 C = 5,
 p0 = 0.1,
 p1 = 0.4,
 power = 0.8,
 alpha = 0.05)
```

`twoarmDesign` *Find two-arm trial designs that use stochastic curtailment*

Description

This function finds admissible design realisations for two-arm binary outcome trials, using stochastic curtailment. The output can be used as the sole argument in the function 'drawDiagram', which will return the stopping boundaries for the admissible design of your choice. Monitoring frequency can set in terms of block size.

Usage

```
twoarmDesign(
  nmin.arm,
  nmax.arm,
  block.size,
  pc,
  pt,
  alpha,
  power,
  maxthetaF = NULL,
  minthetaE = 0.7,
  bounds = "ahern",
  fixed.r = NULL,
  max.combns = 1e+06,
  rm.dominated.designs = TRUE,
  exact.thetaF = NULL,
  exact.thetaE = NULL,
  fast.method = FALSE
)
```

Arguments

`nmin.arm`	Minimum permitted sample size *per arm*. Should be a multiple of block size.
`nmax.arm`	Maximum permitted sample size *per arm*. Should be a multiple of block size.
`block.size`	Block size.
`pc`	Anticipated response rate on the control arm.
`pt`	Anticipated response rate on the treatment arm.
`alpha`	Significance level
`power`	Required power (1-beta).
`maxthetaF`	Maximum value of lower CP threshold theta_F_max.
`minthetaE`	Minimum value of upper threshold theta_E_min.
`bounds`	choose what final rejection boundaries should be searched over: Those of A'Hern ("ahern"), Wald ("wald") or no constraints (NA). Defaults to "wald".

`fixed.r` Choose what final rejection boundaries should be searched over. Useful for reproducing a particular design realisation. Defaults to NULL.

`max.combns` Provide a maximum number of ordered pairs (theta_F, theta_E). Defaults to 1e6.

`rm.dominated.designs` Logical. If TRUE, dominated designs will be removed from final output. Defaults to TRUE.

`exact.thetaF` Provide an exact value for lower threshold theta_F. Useful for reproducing a particular design realisation. Defaults to NULL.

`exact.thetaE` Provide an exact value for upper threshold theta_E. Useful for reproducing a particular design realisation. Defaults to NULL.

`fast.method` Logical. If FALSE, design search is conducted over all combinations of (theta_F, theta_E). If TRUE, a much faster, though less thorough, design search is undertaken. Defaults to FALSE.

Value

Output is a list of two dataframes. The first, $input, is a one-row data frame that contains all the arguments used in the call. The second, $all.des, contains the operating characteristics of all admissible designs found.

Author(s)

Martin Law, <martin.law@mrc-bsu.cam.ac.uk>

Examples

```
des <- twoarmDesign(nmin.arm=20,
nmax.arm=24,
block.size=8,
pc=0.1,
pt=0.4,
alpha=0.1,
power=0.8,
maxthetaF=0.4,
minthetaE=0.7,
max.combns=1e4)
```

Index