Package: seqDesign (via r-universe)

October 29, 2024

Version 1.3

Date 2020-08-13

Title Simulation and Group-Sequential Monitoring of Randomized Treatment Efficacy Trials with Time-to-Event Endpoints

URL https://github.com/mjuraska/seqDesign

Description A broad spectrum of both event-driven and fixed follow-up preventive vaccine efficacy trial designs, including designs of Gilbert, Grove et al. (2011, Statistical Communications in Infectious Diseases), are implemented, with application generally to individual-randomized clinical trials with multiple active treatment groups and a shared control group, and a study endpoint that is a time-to-event endpoint subject to right-censoring. The design accommodates the following features: (1) the possibility that the efficacy of the treatment/vaccine groups may take time to accrue while the multiple treatment administrations/vaccinations are given, (2) hazard ratio and cumulative incidence-based treatment/vaccine efficacy parameters and multiple estimation/hypothesis testing procedures are available, (3) interim/group-sequential monitoring of each treatment group for potential harm, non-efficacy (lack of benefit), efficacy (benefit), and high efficacy, (3) arbitrary alpha spending functions for different monitoring outcomes, (4) arbitrary timing of interim looks, separate for each monitoring outcome, in terms of either event accrual or calendar time, (5) flexible analysis cohort characterization (intention-to-treat vs. per-protocol/as-treated; counting only events for analysis that occur after a specific point in study time), and (6) division of the trial into two stages of time periods where each treatment is first evaluated for efficacy in the first stage of follow-up, and, if and only if it shows significant treatment efficacy in stage one, it is evaluated for longer-term durability of efficacy in stage two. The package produces plots and tables describing operating characteristics of a specified design including a description of monitoring boundaries on

multiple scales for the different outcomes; event accrual since trial initiation; probabilities of stopping early for potential harm, non-efficacy, etc.; an unconditional power for intention-to-treat and per-protocol analyses; calendar time to crossing a monitoring boundary or reaching the target number of endpoints if no boundary is crossed; trial duration; unconditional power for comparing treatment efficacies; and the distribution of the number of endpoints within an arbitrary study time interval (e.g., events occurring after the treatments/vaccinations are given), useful as input parameters for the design of studies of the association of biomarkers with a clinical outcome (surrogate endpoint problem). The code can be used for a single active treatment versus control design and for a single-stage design.

BugReports https://github.com/mjuraska/seqDesign/issues

Depends R (>= 2.16), survival

License GPL-2

Encoding UTF-8

LazyLoad yes

VignetteBuilder knitr, R.rsp

Suggests knitr, R.rsp

RoxygenNote 7.1.1

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

RemoteUrl https://github.com/mjuraska/seqdesign

RemoteRef HEAD

RemoteSha 23965224e807e5512f3a6c75c008ab02f490e9e5

Contents

censTrial *Generation of Pre-Unblinded Follow-Up Data-Sets by Applying the Monitoring Outcomes*

Description

censTrial 'correctly censors' treatment arms in data-sets generated by simTrial by including pre-unblinded follow-up data only according to the monitoring conclusions as reported by monitorTrial.

Usage

```
censTrial(
  dataFile,
  monitorFile,
  stage1,
  stage2,
  saveFile = NULL,
  saveDir = NULL,
  verbose = TRUE
)
```

Arguments

dataFile	if saveDir = NULL, a list returned by simTrial; otherwise a name (character string) of an .RData file created by simTrial
monitorFile	if saveDir = NULL, a list returned by monitorTrial; otherwise a name (character string) of an .RData file created by monitorTrial
stage1	the final week of stage 1 in a two-stage trial
stage2	the final week of stage 2 in a two-stage trial, i.e., the maximum follow-up time
saveFile	a character string specifying the name of the output .RData file. If NULL (default), a default file name will be used.
saveDir	a character string specifying a path for both dataFile and monitorFile. If supplied, the output is also saved as an .RData file in this directory; otherwise the output is returned as a list.
verbose	a logical value indicating whether information on the output directory and file name should be printed out (default is TRUE)

Details

All time variables use week as the unit of time. Month is defined as 52/12 weeks.

The following censoring rules are applied to each data-set generated by simTrial:

- If no vaccine arm registers efficacy or high efficacy in Stage 1, the placebo arm is censored on the date when the last vaccine arm hits the harm or non-efficacy boundary.
- If a vaccine arm hits the harm boundary, censor the arm immediately.

- If a vaccine arm hits the non-efficacy boundary, censor the arm on the earliest date of the two events: (1) the last vaccine arm hits the harm or non-efficacy boundary (if applicable); and (2) all subjects in the vaccine arm have completed the final `stage1` visit.

Value

If `saveDir` is specified, the output list (named `trialListCensor`) is saved as an `.RData` file in `saveDir` (the path to `saveDir` is printed); otherwise it is returned. The output object is a list of length equal to the number of simulated trials, each of which is a `data.frame` with at least the variables `trt`, `entry`, `exit`, and `event` storing the treatment assignments, enrollment times, correctly censored study exit times, and event indicators, respectively. If available, indicators belonging to the per-protocol cohort (named pp1, pp2, etc.) are copied from the uncensored data-sets.

See Also

`simTrial`, `monitorTrial`, and `rankTrial`

Examples

```
simData <- simTrial(N=c(1000, rep(700, 2)), aveVE=seq(0, 0.4, by=0.2),
                    VEmodel="half", vePeriods=c(1, 27, 79), enrollPeriod=78,
                    enrollPartial=13, enrollPartialRelRate=0.5, dropoutRate=0.05,
                    infecRate=0.04, fuTime=156,
                    visitSchedule=c(0, (13/3)*(1:4), seq(13*6/3, 156, by=13*2/3)),
                    missVaccProb=c(0,0.05,0.1,0.15), VEcutoffWeek=26, nTrials=5,
                    stage1=78, randomSeed=300)

monitorData <- monitorTrial(dataFile=simData, stage1=78, stage2=156,
                            harmMonitorRange=c(10,100), alphaPerTest=NULL,
                            nonEffStartMethod="FKG", nonEffInterval=20,
                            lowerVEnoneff=0, upperVEnoneff=0.4, highVE=0.7,
                            stage1VE=0, lowerVEuncPower=0, alphaNoneff=0.05,
                            alphaHigh=0.05, alphaStage1=0.05,
                            alphaUncPower=0.05, estimand="cuminc", lagTime=26)

censData <- censTrial(dataFile=simData, monitorFile=monitorData, stage1=78, stage2=156)

### alternatively, to save the .RData output file (no '<-' needed):
###
### simTrial(N=c(1400, rep(1000, 2)), aveVE=seq(0, 0.4, by=0.2), VEmodel="half",
###          vePeriods=c(1, 27, 79), enrollPeriod=78, enrollPartial=13,
###          enrollPartialRelRate=0.5, dropoutRate=0.05, infecRate=0.04, fuTime=156,
###          visitSchedule=c(0, (13/3)*(1:4), seq(13*6/3, 156, by=13*2/3)),
###          missVaccProb=c(0,0.05,0.1,0.15), VEcutoffWeek=26, nTrials=30,
###          stage1=78, saveDir="./", randomSeed=300)
###
### monitorTrial(dataFile=
###              "simTrial_nPlac=1400_nVacc=1000_1000_aveVE=0.2_0.4_infRate=0.04.RData",
###              stage1=78, stage2=156, harmMonitorRange=c(10,100), alphaPerTest=NULL,
###              nonEffStartMethod="FKG", nonEffInterval=20, lowerVEnoneff=0,
###              upperVEnoneff=0.4, highVE=0.7, stage1VE=0, lowerVEuncPower=0,
###              alphaNoneff=0.05, alphaHigh=0.05, alphaStage1=0.05, alphaUncPower=0.05,
```

```
###              estimand="cuminc", lagTime=26, saveDir="./")
###
### censTrial(dataFile=
###          "simTrial_nPlac=1400_nVacc=1000_1000_aveVE=0.2_0.4_infRate=0.04.RData",
###          monitorFile=
###       "monitorTrial_nPlac=1400_nVacc=1000_1000_aveVE=0.2_0.4_infRate=0.04_cuminc.RData",
###          stage1=78, stage2=156, saveDir="./")
```

`crossBoundCumProb` *Estimate cumulative probabilities of crossing an efficacy or non-efficacy boundary in an event-driven 2-arm trial design*

Description

Computes proportions of simulated trials that crossed either an efficacy or a non-efficacy stopping boundary by analysis $1, \ldots,$ `nAnalyses` using an `.RData` output file from `monitorTrial`. An event-driven 2-arm trial design is assumed.

Usage

```
crossBoundCumProb(
  boundType = c("eff", "nonEff"),
  nAnalyses,
  monitorTrialFile,
  monitorTrialDir = NULL
)
```

Arguments

`boundType`	a character string specifying if the one-sided null hypothesis is of the form $H_0 : \theta \geq \theta_0$ (`"eff"`, default) or $H_0 : \theta \leq \theta_0$ (`"nonEff"`), where θ and θ_0 are the true hazard ratio and its value specifying the null hypothesis, respectively
`nAnalyses`	a numeric value specifying the number of analyses
`monitorTrialFile`	either a character string specifying an `.RData` file or a list outputted by `monitorTrial`
`monitorTrialDir`	a character string specifying a path to `monitorTrialFile` if `monitorTrialFile` specifies a file name

Value

A numeric vector of estimated cumulative probabilities of crossing the specified boundary by analysis $1, \ldots,$ `nAnalyses`.

Examples

```
simData <- simTrial(N=c(1000, 1000), aveVE=c(0, 0.4),
                    VEmodel="half", vePeriods=c(1, 27, 79), enrollPeriod=78,
                    enrollPartial=13, enrollPartialRelRate=0.5, dropoutRate=0.05,
                    infecRate=0.06, fuTime=156, visitSchedule=seq(0, 156, by=4),
                    missVaccProb=0.05, VEcutoffWeek=26, nTrials=5,
                    stage1=78, randomSeed=300)

monitorData <- monitorTrial(dataFile=simData, stage1=78, stage2=156,
                            harmMonitorRange=c(10,75), harmMonitorAlpha=0.05,
                            effCohort=list(timingCohort=list(lagTime=0),
                                           times=c(75, 150),
                                           timeUnit="counts",
                                           lagTime=0,
                                           estimand="cox",
                                           nullVE=0,
                                           nominalAlphas=c(0.001525, 0.024501)),
                            nonEffCohorts=list(timingCohort=list(lagTime=0),
                                               times=c(75, 150),
                                               timeUnit="counts",
                                               cohort1=list(lagTime=0,
                                                            estimand="cox",
                                                            nullVE=0.4,
                                                  nominalAlphas=c(0.001525, 0.024501))),
                            lowerVEnoneff=0, highVE=1, lowerVEuncPower=0,
                            alphaHigh=0.05, alphaUncPower=0.05,
                            verbose=FALSE)

crossBoundCumProb(boundType="eff", nAnalyses=2, monitorTrialFile=monitorData)
crossBoundCumProb(boundType="nonEff", nAnalyses=2, monitorTrialFile=monitorData)
```

`decisionTimes` *Extract the time since trial start to crossing a stopping boundary or reaching the target number of events if no stopping boundary crossed in an event-driven 2-arm trial design*

Description

Obtains times (in weeks) since trial initiation to crossing a harm, non-efficacy or efficacy boundary, or reaching the target number of events if no stopping boundary is crossed in an event-driven 2-arm trial design. The times are extracted from `.RData` files outputted by `monitorTrial`.

Usage

```
decisionTimes(
  monitorTrialFile,
  monitorTrialDir = NULL,
  saveFile = NULL,
```

```
  saveDir = monitorTrialDir
)
```

Arguments

monitorTrialFile	either a character vector specifying (multiple) .RData file(s) or a list of lists outputted by monitorTrial
monitorTrialDir	a character string specifying a path to the file(s) in monitorTrialFile if monitorTrialFile specifies file name(s)
saveFile	a character string optionally specifying an .RData file name storing the output (NULL by default)
saveDir	a character string optionally specifying the file path for saveFile (set to monitorTrialDir by default)

Value

A list (of the same length as monitorTrialFile) of numeric vectors of times. The order of the vectors matches the order of components in monitorTrialFile. The length of each vector equals the number of simulated trials in the corresponding output list from monitorTrial.

Examples

```
simData <- simTrial(N=c(1000, 1000), aveVE=c(0, 0.4),
                    VEmodel="half", vePeriods=c(1, 27, 79), enrollPeriod=78,
                    enrollPartial=13, enrollPartialRelRate=0.5, dropoutRate=0.05,
                    infecRate=0.6, fuTime=156,
                    visitSchedule=c(0, (13/3)*(1:4), seq(13*6/3, 156, by=13*2/3)),
                    missVaccProb=0.05, VEcutoffWeek=26, nTrials=5,
                    stage1=78, randomSeed=300)

monitorData <- monitorTrial(dataFile=simData, stage1=78, stage2=156,
                            harmMonitorRange=c(10,75), harmMonitorAlpha=0.05,
                            effCohort=list(timingCohort=list(lagTime=0),
                                           times=c(75, 150),
                                           timeUnit="counts",
                                           lagTime=0,
                                           estimand="cox",
                                           nullVE=0,
                                           nominalAlphas=c(0.001525, 0.024501)),
                            nonEffCohorts=list(timingCohort=list(lagTime=0),
                                               times=c(75, 150),
                                               timeUnit="counts",
                                               cohort1=list(lagTime=0,
                                                            estimand="cox",
                                                            nullVE=0.4,
                                                  nominalAlphas=c(0.001525, 0.024501))),
                            lowerVEnoneff=0, highVE=1, lowerVEuncPower=0,
                            alphaHigh=0.05, alphaUncPower=0.05,
                            verbose=FALSE)
```

```
times <- decisionTimes(list(monitorData))

## alternatively, to save the .RData output file (no '<-' needed):
## decisionTimes(list(monitorData), saveFile="decisionTimes.RData")
```

`estHRbound` *Estimate hazard ratios at an efficacy or non-efficacy stopping boundary defined using the Wald CI approach in an event-driven 2-arm trial design*

Description

Assuming an exponential survival model, hazard ratios are estimated at an efficacy or non-efficacy stopping boundary, defined using the Wald CI approach, at each group-sequential analysis in an event-driven 2-arm trial design.

Usage

```
estHRbound(boundType = c("eff", "nonEff"), nullHR, alpha, nEvents, randFrac)
```

Arguments

`boundType`	a character string specifying if the one-sided null hypothesis is of the form $H_0 : \theta \geq \theta_0$ (`"eff"`, default) or $H_0 : \theta \leq \theta_0$ (`"nonEff"`), where θ is the hazard ratio and θ_0 is specified by `nullHR`
`nullHR`	a nonnegative numeric value specifying the hazard ratio, θ_0, under the null hypothesis. If the null hypothesis differs across multiple analyses, `nullHR` may be a numeric vector of equal length as `alpha`.
`alpha`	a numeric vector of two-sided nominal significance levels (e.g., those defined by the O'Brien-Fleming group-sequential test)
`nEvents`	a numeric vector of numbers of events at which analyses are performed. The lengths of `alpha` and `nEvents` must be the same, and the components of the two vectors must correspond to each other.
`randFrac`	a fraction of subjects randomized to the group considered in the hazard ratio's numerator

Details

Using an exponential survival model and sample estimates $\widehat{\lambda}_1$ and $\widehat{\lambda}_2$ of the group-specific hazard rates, the asymptotic variance of the log hazard ratio estimator $\log \widehat{\theta} = \log(\widehat{\lambda}_1/\widehat{\lambda}_2)$ is employed together with the approximation $E\{\delta|\lambda_1\} = (\widehat{\lambda}_1/\widehat{\lambda}_2)\, E\{\delta|\lambda_2\}$. The resultant variance approximation is $\mathrm{var}\{\log \widehat{\theta}\} = (1/D)\{2 + p\,\widehat{\theta}/(1-p) + (1-p)/(p\,\widehat{\theta})\}$, where D is the arm-pooled number of events `nEvents` and p is the randomization fraction `randFrac`.

Value

A data frame (with rows corresponding to the components of `alpha` and `nEvents`) of point estimates of the hazard ratio at the stopping boundary and the pertaining monitoring-adjusted $(1-\alpha^*)\times 100\%$ confidence intervals, where α^* is the overall two-sided type 1 error rate.

Examples

```
## O'Brien-Fleming test of H0: HR >= 0.7 (for efficacy) at
## 35%, 70%, and 100% of the total information under 1:1 randomization
estHRbound("eff", nullHR=0.7, alpha=c(0.00030, 0.01466, 0.04548),
           nEvents=c(53, 106, 151), randFrac=0.5)

## O'Brien-Fleming test of H0: HR <= 0.5 (for non-efficacy) at
## 35%, 70%, and 100% of the total information under 1:1 randomization
estHRbound("nonEff", nullHR=0.5, alpha=c(0.00030, 0.01466, 0.04548),
           nEvents=c(53, 106, 151), randFrac=0.5)
```

---

getBlockSize *Determine block size for use in blocked randomization*

Description

`getBlockSize` returns the minimum block size (possibly within a specified range) that is compatible with a trial's overall treatment assignment totals.

Usage

```
getBlockSize(nvec, range = c(0, Inf))
```

Arguments

`nvec`	vector specifying the number of participants to be assigned to each treatment group. The vector should have one component per group, so that its length equals number of groups. The sum of `nvec` should equal the total enrollment for the trial.
`range`	(Optional) vector of length two giving the lower and upper bounds (respectively) on block sizes that the user wishes to consider.

Details

The ordering of the components of `nvec` is not important, so using `nvec = c(x,y,z)` will produce the same results as using `nvec = c(z,x,y)`.

In block randomization one does not necessarily want the smallest block size, which is the reason for the existance of the `range` argument. For example, a trial with a 1:1 randomization allocation between two groups would have a minimum block size of 2, which most people would consider to be too small. So a typical usage of `getBlockSize` would be to use `range` to set a minimum

acceptable block size, through use of vector of form `c(lowerBound, Inf)`. A large trial should probably have a block size on the order of 10-20 or larger, depending on factors including the total trial size and speed of enrollment, so setting a minimum is a good idea.

Value

An integer or NA. If the user does not specify `range`, then the function will always return an integer, which is the smallest block size compatible with the specified vector of treatment group sizes. If the user *has* specified the `range`, then the function adds the further constraint that the block size must lie in the closed interval given by `range` (i.e., the block size must be greater-than-or-equal-to `range[1]` and less-than-or-equal-to `range[2]`). If there are no compatible block sizes that lie in the given interval, then an NA is returned.

Note that the value returned is the **minimum** block size that is compatible, not necessarily the only one. Any other compatible block sizes (if any exist) will be integer multiples of the minimum size. You can check the feasibility of various integer multiples by seeing if they divide evenly into the total trial size (i.e., into the sum of `nvec`).

Examples

```
getBlockSize(nvec = c(375, 375) )
## specify a minimum block size of 10 (no maximum)
getBlockSize(nvec = c(375, 375), range = c(10, Inf) )

getBlockSize( nvec = c(30, 510, 390) )
## require a minimum block size of 10 and maximum of 30
## (not possible with this nvec, so function returns NA)
getBlockSize( nvec = c(30, 510, 390), range = c(10, 30) )
```

---

monitorTrial *Group Sequential Monitoring of Simulated Efficacy Trials for the Event of Potential Harm, Non-Efficacy, and High Efficacy*

Description

`monitorTrial` applies a group sequential monitoring procedure to data-sets generated by `simTrial`, which may result in modification or termination of each simulated trial.

Usage

```
monitorTrial(
  dataFile,
  stage1,
  stage2,
  harmMonitorRange,
  harmMonitorAlpha = 0.05,
  alphaPerTest = NULL,
  nonEffStartMethod = c("FKG", "fixed"),
```

```
  nonEffStartParams = NULL,
  nonEffIntervalUnit = c("counts", "time"),
  nonEffInterval,
 nonEffCohorts = list(times = NULL, timeUnit = "counts", timingCohort = list(lagTime =
   NULL, cohortInd = NULL), cohort1 = list(estimand = "cox", lagTime = NULL, cohortInd =
     NULL, nullVE = NULL, nominalAlphas = NULL)),
 effCohort = list(times = NULL, timeUnit = "counts", timingCohort = list(lagTime =
   NULL, cohortInd = NULL), nullVE = NULL, estimand = "cox", lagTime = NULL, cohortInd =
     NULL, nominalAlphas = NULL),
  stage1Eff = list(cohort = list(lagTime = 0, cohortInd = NULL), nullVE = NULL,
     nominalAlpha = NULL, estimand = "cox"),
  lowerVEnoneff = NULL,
  highVE,
  stage1VE,
  lowerVEuncPower = NULL,
  alphaHigh,
  alphaStage1,
  alphaUncPower = NULL,
  saveFile = NULL,
  saveDir = NULL,
  verbose = TRUE
)
```

Arguments

dataFile	if saveDir = NULL, a list returned by simTrial; otherwise a name (character string) of an .RData file created by simTrial
stage1	the final week of stage 1 in a two-stage fixed-follow-up trial
stage2	the final week of stage 2 in a two-stage fixed-follow-up trial, i.e., the maximum total follow-up time
harmMonitorRange	a 2-component numeric vector specifying the range of the pooled number of events (pooled over the placebo and vaccine arm accruing events the fastest) over which the type I error rate, specified in harmMonitorAlpha, shall be spent (per vaccine arm). Note that harmMonitorRange does not specify a range over which potential-harm stopping boundaries will be computed; instead, it specifies when potential-harm monitoring will start, and the range over which the type I error rate harmMonitorAlpha will be spent. If nonEffStartMethod = "FKG" or "fixed", then the second value is ignored and can be replaced with NA.
harmMonitorAlpha	a numeric value (0.05 by default) specifying the overall type I error rate for potential-harm monitoring (per vaccine arm). To turn off potential-harm monitoring, set harmMonitorAlpha equal to 0.00001.
alphaPerTest	a per-test nominal significance level for potential-harm monitoring. If NULL (default), a per-test significance level is calculated that yields the overall type I error rate of harmMonitorAlpha at the end of harmMonitorRange.

nonEffStartMethod

a character string specifying the method used for determining when non-efficacy monitoring is to start. The default method of Freidlin, Korn, and Gray (2010) ("FKG") calculates the minimal pooled event count (pooled over the placebo and vaccine arm accruing events the fastest) such that a hazard-ratio-based VE point estimate of 0% would result in declaring non-efficacy, i.e., the upper bound of the two-sided $(1 - \texttt{alphaNoneff})100\%$ confidence interval for VE based on the asymptotic variance of the log-rank statistic equals the non-efficacy threshold specified as component upperVEnonEff in the list nonEffStartParams. If this list component is left unspecified, the argument upperVEnonEff is used as the non-efficacy threshold. The alternative method ("fixed") starts non-efficacy monitoring at a fixed pooled event count (pooled over the placebo and vaccine arm accruing events the fastest) specified by component N1 in the list nonEffStartParams.

nonEffStartParams

a list with named components specifying parameters required by nonEffStartMethod (NULL by default)

nonEffIntervalUnit

a character string specifying whether intervals between two adjacent non-efficacy interim analyses should be event-driven (default option "counts") or calendar time-driven (option "time")

nonEffInterval a numeric vector (a number of events or a number of weeks) specifying the timing of non-efficacy interim analyses. If a single numeric value is specified, then all interim looks are equidistant (in terms of the number of events or weeks), and the value specifies the constant increment of information or time between two adjacent interim looks. If a numeric vector with at least two components is specified, then, following the initial interim look, the timing of subsequent interim looks is determined by (potentially differential) increments of information or time specified by this vector.

nonEffCohorts a named list specifying all cohorts (for both timing and analysis) used in non-efficacy monitoring. The required list components characterize the 'timing cohort,' i.e., the cohort events in which determine the analysis timepoints in an event-driven design (components times, timeUnit, and timingCohort), and the analysis cohort(s), i.e., the cohort(s) in which inference is made about the null hypothesis of non-efficacy (component cohort1 is required, and cohort2, etc. are optional for additional analysis cohorts if, e.g., non-efficacy monitoring is conducted in both the ITT and per-protocol cohorts). As for the timing cohort, times specifies the analysis timepoints in terms of event counts (timeUnit = "counts" is the only implemented option). timingCohort is a list characterizing the timing cohort by components lagTime, a lag time (in weeks) that controls event inclusion for timing (if no lag is desired, it can be set to NULL or 0), and cohortInd, a character string naming an indicator variable included in the data frames outputted by simTrial, which also controls event inclusion for the purpose of determining analysis timepoints. The other top-level list components cohort1, cohort2, etc. are each a list that must contain a component named estimand, which can take on values "cox", "cuminc", or "combined". Optional list components in cohort1, cohort2, etc. are lagTime, cohortInd, nullVE, and nominalAlphas. lagTime specifies a lag time (in

weeks) that controls event inclusion for the analysis cohort. If no lag time is desired, then this component can be ignored, set to NULL, or set to 0. cohortInd is a character string naming an indicator variable included in the data frames outputted by simTrial to be used to subset participants into the analysis cohort. This component allows inclusion of, e.g., a "per-protocol" variable (e.g., by setting cohortInd to "pp1"). nullVE specifies the one-sided null hypothesis as $H_0 : VE \geq$ nullVE. The rejection of H_0 constitutes evidence for non-efficacy. nominalAlphas specifies nominal significance levels in a two-arm event-driven trial design.

effCohort — a named list specifying the timing and analysis cohorts used in group-sequential efficacy monitoring (if part of the monitoring plan). The required list components characterize the 'timing cohort,' i.e., the cohort events in which determine the analysis timepoints in an event-driven design (components times, timeUnit, and timingCohort), and a single analysis cohort, i.e., the cohort in which inference is made about the null hypothesis of efficacy (components estimand, lagTime, cohortInd, nullVE, and nominalAlphas). As for the timing cohort, times specifies the analysis timepoints in terms of event counts (timeUnit $=$ "counts" is the only implemented option). timingCohort is a list characterizing the timing cohort by components lagTime, a lag time (in weeks) that controls event inclusion for timing (if no lag is desired, it can be set to NULL or 0), and cohortInd, a character string naming an indicator variable included in the data frames outputted by simTrial, which also controls event inclusion for the purpose of determining analysis timepoints. As for the analysis cohort, estimand can take on values "cox", "cuminc", or "combined". lagTime specifies a lag time (in weeks) that controls event inclusion for the analysis cohort. If no lag time is desired, then this component can be ignored, set to NULL, or set to 0. cohortInd is a character string naming an indicator variable included in the data frames outputted by simTrial to be used to subset participants into the analysis cohort. This component allows inclusion of, e.g., a "per-protocol" variable (e.g., by setting cohortInd to "pp1"). nullVE specifies the one-sided primary null hypothesis as $H_0 : VE \leq$ nullVE. The rejection of H_0 constitues evidence for clinically relevant efficacy. nominalAlphas specifies nominal significance levels in a two-arm event-driven trial design.

lowerVEnoneff — specifies an additional criterion for declaring non-efficacy if the hypothesis test is based on Wald confidence interval(s). It requires that the lower bound of the two-sided Wald CI(s) for the VE estimand(s), at the confidence level determined by nonEffCohorts$cohort1$nominalAlphas, etc., lie(s) below lowerVEnoneff (typically set equal to 0). If NULL (default), this criterion is ignored.

highVE — specifies a criterion for declaring high-efficacy: the lower bound of the two-sided $(1 - \text{alphaHigh})100\%$ confidence interval for the VE estimand lies above highVE (typically a number in the 0.5–1 range). To turn off high efficacy monitoring, set highVE equal to 1.

stage1VE — specifies a criterion for advancement of a treatment's evaluation into Stage 2: the lower bound of the two-sided $(1 - \text{alphaStage1})100\%$ confidence interval for the VE estimand lies above stage1VE (typically set equal to 0)

lowerVEuncPower — a numeric vector with each component specifying a one-sided null hypothesis $H_0 : VE(0 - \text{stage1}) \leq \text{lowerVEuncPower} \times 100\%$. Unconditional

power (i.e., accounting for sequential monitoring) to reject each H_0 is calculated, where the rejection region is defined by the lower bound of the two-sided $(1-$`alphaUncPower`$)100\%$ confidence interval for the VE estimand being above the respective component of `lowerVEuncPower` (typically values in the 0–0.5 range).

`alphaHigh` one minus the nominal confidence level of the two-sided confidence interval used for high efficacy monitoring

`alphaStage1` one minus the nominal confidence level of the two-sided confidence interval used for determining whether a treatment's evaluation advances into Stage 2

`alphaUncPower` one minus the nominal confidence level of the two-sided confidence interval used to test one-sided null hypotheses $H_0 : VE(0-$`stage1`$) \leq$ `lowerVEuncPower`$\times 100\%$ against alternative hypotheses $H_1 : VE(0-$`stage1`$) >$ `lowerVEuncPower`$\times 100\%$. The same nominal confidence level is applied for each component of `lowerVEuncPower`.

`saveFile` a character string specifying the name of the output `.RData` file. If `NULL` (default), a default file name will be used.

`saveDir` a character string specifying a path for `dataFile`. If supplied, the output is also saved as an `.RData` file in this directory; otherwise the output is returned as a list.

`verbose` a logical value indicating whether information on the output directory, file name, and monitoring outcomes should be printed out (default is `TRUE`)

Details

All time variables use week as the unit of time. Month is defined as 52/12 weeks.

Potential harm monitoring starts at the `harmMonitorRange[1]`-th infection pooled over the placebo group and the vaccine regimen that accrues infections the fastest. The potential harm analyses continue at each additional infection until the first interim analysis for non-efficacy. The monitoring is implemented with exact one-sided binomial tests of H0: $p \leq p0$ versus H1: $p > p0$, where p is the probability that an infected participant was assigned to the vaccine group, and $p0$ is a fixed constant that represents the null hypothesis that an infection is equally likely to be assigned vaccine or placebo. Each test is performed at the same prespecified nominal/unadjusted alpha-level (`alphaPerTest`), chosen based on simulations such that, for each vaccine regimen, the overall type I error rate by the `harmMonitorRange[2]`-th arm-pooled infection (i.e., the probability that the potential harm boundary is reached when the vaccine is actually safe, $p = p0$) equals `harmMonitorAlpha`.

Non-efficacy is defined as evidence that it is highly unlikely that the vaccine has a beneficial effect measured as VE(0–`stage1`) of `upperVEnoneff` x 100% or more. The non-efficacy analyses for each vaccine regimen will start at the first infection (pooled over the vaccine and placebo arm) determined by `nonEffStartMethod`. Stopping for non-efficacy will lead to a reported two-sided (1-`alphaNoneff`) x 100% CI for VE(0–`stage1`) with, optionally, the lower confidence bound below `lowerVEnoneff` and the upper confidence bound below `upperVEnoneff`, where `estimand` determines the choice of the VE(0–`stage1`) estimand. This approach is similar to the inefficacy monitoring approach of Freidlin, Korn, and Gray (2010). If `estimand = "combined"`, stopping for non-efficacy will lead to reported (1-`alphaNoneff`) x 100% CIs for both VE parameters with, optionally, lower confidence bounds below `lowerVEnoneff` and upper confidence bounds

below upperVEnoneff. If laggedMonitoring = TRUE, stopping for non-efficacy will lead to reported (1-alphaNoneff) x 100% CIs for both VE(0–stage1) and VE(lagTime–stage1) with, optionally, lower confidence bounds below lowerVEnoneff and upper confidence bounds below upperVEnoneff.

High efficacy monitoring allows early detection of a highly protective vaccine if there is evidence that VE(0–stage2) $>$ highVE x 100%. It is synchronized with non-efficacy monitoring during Stage 1, and a single high-efficacy interim analysis during Stage 2 is conducted halfway between the end of Stage 1 and the end of the trial. While monitoring for potential harm and non-efficacy restricts to stage1 infections, monitoring for high efficacy counts all infections during stage1 or stage2, given that early stopping for high efficacy would only be warranted under evidence for durability of the efficacy.

The following principles and rules are applied in the monitoring procedure:

- Exclude all follow-up data from the analysis post-unblinding (and include all data pre-unblinding).
- The monitoring is based on modified ITT analysis, i.e., all subjects documented to be free of the study endpoint at baseline are included and analyzed according to the treatment assigned by randomization, ignoring how many vaccinations they received (only pre-unblinding follow-up included).
- If a vaccine hits the harm boundary, immediately discontinue vaccinations and accrual into this vaccine arm, and unblind this vaccine arm (continue post-unblinded follow-up until the end of Stage 1 for this vaccine arm).
- If a vaccine hits the non-efficacy boundary, immediately discontinue vaccinations and accrual into this vaccine arm, keep blinded and continue follow-up until the end of Stage 1 for this vaccine arm.
- If and when the last vaccine arm hits the non-efficacy (or harm) boundary, discontinue vaccinations and accrual into this vaccine arm, and unblind (the trial is over, completed in Stage 1).
- Stage 1 for the whole trial is over on the earliest date of the two events: (1) all vaccine arms have hit the harm or non-efficacy boundary; and (2) the last enrolled subject in the trial reaches the final stage1 visit.
- Continue blinded follow-up until the end of Stage 2 for each vaccine arm that reaches the end of stage1 with a positive efficacy (as defined by stage1VE) or high efficacy (as defined by highVE) result.
- If at least one vaccine arm reaches the end of stage1 with a positive efficacy or high efficacy result, continue blinded follow-up in the placebo arm until the end of Stage 2.
- Stage 2 for the whole trial is over on the earliest date of the two events: (1) all subjects in the placebo arm and each vaccine arm that registered efficacy or high efficacy in stage1 have failed or been censored; and (2) all subjects in the placebo arm and each vaccine arm that registered efficacy or high efficacy in stage1 have completed the final stage2 visit.

The above rules have the following implications:

- If a vaccine hits the non-efficacy boundary but Stage 1 for the whole trial is not over, then one includes in the analysis all follow-up through the final stage1 visit for that vaccine regimen, including all individuals accrued up through the date of hitting the non-efficacy boundary (which will be the total number accrued to this vaccine arm).

- If a vaccine hits the harm boundary, all follow-up information through the date of hitting the harm boundary is included for this vaccine; no follow-up data are included after this date.
- If and when the last vaccine arm hits the non-efficacy (or harm) boundary, all follow-up information through the date of hitting the non-efficacy (or harm) boundary is included for this vaccine; no follow-up data are included after this date.

Value

If `saveDir` (and, optionally `saveFile`) is specified, the output list (named `out`) is saved as an `.RData` file in `saveDir` (the path to `saveDir` is printed); otherwise it is returned. The output object is a list of length equal to the number of simulated trials, each of which is a list of length equal to the number of treatment arms, each of which is a list with (at least) the following components:

- `boundHit`: a character string stating the monitoring outcome in this treatment arm, i.e., one of `"Harm"`, `"NonEffInterim"`, `"NonEffFinal"`, `"Eff"`, or `"HighEff"`. The first four outcomes can occur in Stage 1, whereas the last outcome can combine data over Stage 1 and Stage 2.
- `stopTime`: the time of hitting a stopping boundary since the first subject enrolled in the trial
- `stopInfectCnt`: the pooled number of infections at `stopTime`
- `summObj`: a `data.frame` containing summary information from each non-/high efficacy interim analysis
- `finalHRci`: the final CI for the hazard ratio, available if `estimand!="cuminc"` and there is at least 1 infection in each arm
- `firstNonEffCnt`: the number of infections that triggered non-efficacy monitoring (if available)
- `totInfecCnt`: the total number of `stage1` (`stage2` if `boundHit = "HighEff"`) infections
- `totInfecSplit`: a table with the numbers of `stage1` (`stage2` if `boundHit = "HighEff"`) infections in the treatment and control arm
- `lastExitTime`: the time between the first subject's enrollment and the last subject's exiting from the trial

References

Freidlin B., Korn E. L., and Gray R. (2010), A general inefficacy interim monitoring rule for randomized clinical trials. *Clinical Trials* 7(3):197-208.

See Also

`simTrial`, `censTrial`, `rankTrial`, `estHRbound`, `crossBoundCumProb`, and `decisionTimes`

Examples

```
simData <- simTrial(N=c(1000, 1000), aveVE=c(0, 0.4),
                    VEmodel="half", vePeriods=c(1, 27, 79), enrollPeriod=78,
                    enrollPartial=13, enrollPartialRelRate=0.5, dropoutRate=0.05,
                    infecRate=0.06, fuTime=156, visitSchedule=seq(0, 156, by=4),
                    missVaccProb=0.05, VEcutoffWeek=26, nTrials=5,
                    stage1=78, randomSeed=300)
```

```
### trial design: fixed follow-up; cumulative incidence-based VE estimand;
                  no efficacy monitoring; non-efficacy monitoring using
                  the Freidlin et al. method; hypothesis tests based on Wald
                  confidence intervals
### RIGHT NOW, THE BELOW CALL IS BROKEN
monitorData <- monitorTrial(dataFile=simData, stage1=78, stage2=156,
                            harmMonitorRange=c(10, NA), harmMonitorAlpha=0.05,
                            nonEffStartMethod="FKG", nonEffInterval=20,
                            nonEffCohorts=list(timeUnit="counts",
                                               # right now 'timingCohort' is required
                                               timingCohort=list(lagTime=0),
                                               cohort1=list(estimand="cuminc",
                                                            nullVE=0.4,
                                                            nominalAlphas=0.025)),
                            # it appears that right now 'effCohort' must be specified
                            # even when there is no efficacy monitoring;
                            # this call of monitorTrial() still doesn't run
                            # because it requires event counts for timing
                            effCohort=list(timeUnit="counts",
                                           timingCohort=list(lagTime=0)),
                            lowerVEnoneff=0, highVE=0.7, lowerVEuncPower=0,
                            alphaHigh=0.05, alphaUncPower=0.05)

### trial design: event-driven; hazard-based VE estimand;
                  harmonized efficacy and non-efficacy monitoring;
                  hypothesis tests using the score test in the Cox model
### THE SCORE TEST OPTION NEEDS TO BE ADDED
monitorData <- monitorTrial(dataFile=simData, stage1=78, stage2=156,
                            harmMonitorRange=c(10, 50), harmMonitorAlpha=0.05,
                            nonEffCohorts=list(
                              times=c(50, 100, 150),
                              timeUnit="counts",
                              timingCohort=list(lagTime=0),
                              cohort1=list(estimand="cox",
                                           nullVE=0.4,
                                           nominalAlphas=c(0.0001, 0.0060, 0.0231))),
                            effCohort=list(times=c(50, 100, 150),
                                           timeUnit="counts",
                                           timingCohort=list(lagTime=0),
                                           estimand="cox",
                                           nullVE=0,
                                           nominalAlphas=c(0.0001, 0.0060, 0.0231)),
                            highVE=1, lowerVEuncPower=0,
                            alphaHigh=0.05, alphaUncPower=0.05)

### alternatively, to save the .RData output file (no '<-' needed):
###
### simTrial(N=c(1000, 1000), aveVE=c(0, 0.4),
###          VEmodel="half", vePeriods=c(1, 27, 79), enrollPeriod=78,
###          enrollPartial=13, enrollPartialRelRate=0.5, dropoutRate=0.05,
###          infecRate=0.06, fuTime=156, visitSchedule=seq(0, 156, by=4),
###          missVaccProb=0.05, VEcutoffWeek=26, nTrials=5,
```

```
###          stage1=78, saveDir="./", randomSeed=300)
###
### THIS CALL NEEDS TO BE REVISED TO A SIMPLE BUT WORKING CODE
### THE INTENT HERE IS ONLY TO ILLUSTRATE HOW OUTPUT FILES CAN BE READ IN
### THE PURPOSE IS NOT TO SHOW ANY ADDITIONAL TRIAL DESIGNS
### monitorTrial(dataFile=
###              "simTrial_nPlac=1400_nVacc=1000_1000_aveVE=0.2_0.4_infRate=0.04.RData",
###              stage1=78, stage2=156, harmMonitorRange=c(10,100), alphaPerTest=NULL,
###              nonEffStartMethod="FKG", nonEffInterval=20, lowerVEnoneff=0,
###              upperVEnoneff=0.4, highVE=0.7, stage1VE=0, lowerVEuncPower=0,
###              alphaNoneff=0.05, alphaHigh=0.05, alphaStage1=0.05, alphaUncPower=0.05,
###              estimand="cuminc", lagTime=26, saveDir="./")
```

`rankTrial` *Ranking and Selection, and Head-to-Head Comparison of Individual and Pooled Treatment Arms*

Description

`rankTrial` assesses the probability of correctly selecting the winning most efficacious (individual and/or pooled) treatment arm, and assesses power to detect relative treatment efficacy in head-to-head comparisons of (individual and/or pooled) treatment arms.

Usage

```
rankTrial(
  censFile,
  idxHighestVE,
  headHead = NULL,
  poolHead = NULL,
  lowerVE,
  stage1,
  stage2,
  alpha,
  saveDir = NULL,
  verbose = TRUE
)
```

Arguments

`censFile`	if `saveDir = NULL`, a list returned by `censTrial`; otherwise a name (character string) of an `.RData` file created by `censTrial`
`idxHighestVE`	an integer value identifying the treatment (vaccine) arm with the true highest VE(0–`stage2`)
`headHead`	a matrix (`ncol = 2`) of treatment arm indices for head-to-head comparisons, where the treatment with higher efficacy is listed first in each row

poolHead	a matrix (ncol equals 3 or 4) of treatment arm indices for pooled-arm comparisons, where the pooled treatment with higher efficacy pooled over the first two columns is compared with the (pooled) treatment defined by columns 3 and onward. Ranking and selection of pooled arms is performed separately for each row of poolHead.
lowerVE	a numeric value defining a 'winning' treatment arm as one with sufficient evidence for rejecting the null hypothesis H0: VE(0–stage1) $\leq$ lowerVE x 100% (typically set equal to 0)
stage1	the final week of stage 1 in a two-stage trial
stage2	the final week of stage 2 in a two-stage trial, i.e., the maximum follow-up time
alpha	one minus the nominal confidence level of the two-sided confidence interval used for testing a null hypothesis H0: VE(0–stage1) $\leq b$ x 100% against an alternative hypothesis H1: VE(0–stage1) $> b$ x 100%
saveDir	a character string specifying a path for censFile. If supplied, the output is also saved as an .RData file in this directory; otherwise the output is returned as a list.
verbose	a logical value indicating whether information on the output directory and file name should be printed out (default is TRUE)

Details

All time variables use week as the unit of time. Month is defined as 52/12 weeks.

The probability of correct treatment selection is defined as the probability that the treatment arm with the highest estimated VE(0–stage2) is the one with the true highest VE(0–stage2) and, for this treatment arm, the null hypothesis H0: VE(0–stage1) $\leq$ lowerVE x 100% is rejected. If poolHead is specified, the probability of correct pooled treatment selection is assessed for each set of two pooled treatment arms.

VE(0–t) is estimated as one minus the ratio of Nelson-Aalen-based cumulative incidence functions. The null hypothesis H0: VE(0–t) $\leq b$ x 100% is rejected if the lower bound of the two-sided (1-alpha) x 100% confidence interval for VE(0–t) lies above b.

For head-to-head individual and pooled treatment comparisons, powers to reject the null hypotheses that relative VE(0–stage1) $\leq$ 0% and relative VE(0–stage2) $\leq$ 0% are assessed using the aforementioned testing rule.

Value

If saveDir is specified, the output list (named out) is saved as an .RData file in saveDir (the path to saveDir is printed); otherwise it is returned. The output object is a list with the following components:

- rankSelectPw: the probability of correct selection of the winning most efficacious individual treatment
- headHeadPw: if headHead is specified, a matrix of powers to detect relative VE(0–stage1) (column 1) and relative VE(0–stage2) (column 2) in head-to-head comparisons of individual treatment arms

- poolRankSelectPw: if poolHead is specified, a numeric vector of the probabilities of correct selection of the winning most efficacious pooled treatment for each set of pooled treatments
- poolHeadPw: if poolHead is specified, a matrix of powers to detect relative VE(0–stage1) (column 1) and relative VE(0–stage2) (column 2) in head-to-head comparisons of pooled treatment arms

See Also

simTrial, monitorTrial, and censTrial

Examples

```
simData <- simTrial(N=c(1000, rep(700, 2)), aveVE=seq(0, 0.4, by=0.2),
                    VEmodel="half", vePeriods=c(1, 27, 79), enrollPeriod=78,
                    enrollPartial=13, enrollPartialRelRate=0.5, dropoutRate=0.05,
                    infecRate=0.04, fuTime=156,
                    visitSchedule=c(0, (13/3)*(1:4), seq(13*6/3, 156, by=13*2/3)),
                    missVaccProb=c(0,0.05,0.1,0.15), VEcutoffWeek=26, nTrials=5,
                    stage1=78, randomSeed=300)

monitorData <- monitorTrial(dataFile=simData, stage1=78, stage2=156,
                            harmMonitorRange=c(10,100), alphaPerTest=NULL,
                            nonEffStartMethod="FKG", nonEffInterval=20,
                            lowerVEnoneff=0, upperVEnoneff=0.4,
                            highVE=0.7, stage1VE=0, lowerVEuncPower=0,
                            alphaNoneff=0.05, alphaHigh=0.05, alphaStage1=0.05,
                            alphaUncPower=0.05, estimand="cuminc", lagTime=26)

censData <- censTrial(dataFile=simData, monitorFile=monitorData, stage1=78, stage2=156)

rankData <- rankTrial(censFile=censData, idxHighestVE=2,
                      headHead=matrix(2:1, nrow=1, ncol=2), lowerVE=0, stage1=78,
                      stage2=156, alpha=0.05)

### alternatively, to save the .RData output file (no '<-' needed):
###
### simTrial(N=c(1400, rep(1000, 2)), aveVE=seq(0, 0.4, by=0.2), VEmodel="half",
###          vePeriods=c(1, 27, 79), enrollPeriod=78, enrollPartial=13,
###          enrollPartialRelRate=0.5, dropoutRate=0.05, infecRate=0.04, fuTime=156,
###          visitSchedule=c(0, (13/3)*(1:4), seq(13*6/3, 156, by=13*2/3)),
###          missVaccProb=c(0,0.05,0.1,0.15), VEcutoffWeek=26, nTrials=30,
###          stage1=78, saveDir="./", randomSeed=300)
###
### monitorTrial(dataFile=
###          "simTrial_nPlac=1400_nVacc=1000_1000_aveVE=0.2_0.4_infRate=0.04.RData",
###          stage1=78, stage2=156, harmMonitorRange=c(10,100), alphaPerTest=NULL,
###          nonEffStartMethod="FKG", nonEffInterval=20,
###          lowerVEnoneff=0, upperVEnoneff=0.4, highVE=0.7, stage1VE=0,
###          lowerVEuncPower=0, alphaNoneff=0.05, alphaHigh=0.05, alphaStage1=0.05,
###          alphaUncPower=0.05, estimand="cuminc", lagTime=26, saveDir="./")
###
### censTrial(dataFile=
```

```
###  "simTrial_nPlac=1400_nVacc=1000_1000_aveVE=0.2_0.4_infRate=0.04.RData",
###  monitorFile=
###  "monitorTrial_nPlac=1400_nVacc=1000_1000_aveVE=0.2_0.4_infRate=0.04_cuminc.RData",
###  stage1=78, stage2=156, saveDir="./")
###
### rankTrial(censFile=
###  "trialDataCens_nPlac=1400_nVacc=1000_1000_aveVE=0.2_0.4_infRate=0.04_cuminc.RData",
###  idxHighestVE=2, headHead=matrix(2:1, nrow=1, ncol=2), lowerVE=0, stage1=78,
###  stage2=156, alpha=0.05, saveDir="./")
```

`simTrial` *Simulation of Multi-Arm Randomized Phase IIb/III Efficacy Trials with Time-to-Event Endpoints*

Description

`simTrial` generates independent time-to-event data-sets according to a user-specified trial design. The user makes assumptions about the enrollment, dropout, and infection processes in each treatment arm.

Usage

```
simTrial(
  N,
  VEmodel = c("half", "constant", "manual"),
  aveVE = NULL,
  vePeriods,
  veByPeriod = NULL,
  enrollPeriod,
  enrollPartial,
  enrollPartialRelRate,
  dropoutRate,
  infecRate,
  fuTime,
  visitSchedule,
  missVaccProb = NULL,
  VEcutoffWeek,
  nTrials,
  blockSize = NULL,
  stage1,
  saveFile = NULL,
  saveDir = NULL,
  verbose = TRUE,
  randomSeed = NULL
)
```

Arguments

N	a numeric vector specifying the numbers of enrolled trial participants per treatment arm. The length of N equals the total number of treatment arms, and the first component of N represents the control arm. #' @param VEmodel a character string specifying whether VE is assumed to be constant throughout the follow-up period (option "constant"), have two or three levels over time specified by aveVE (option "half", default), or have multiple levels fully specified by veByPeriod (option "manual"). The option "half" allows either a 2- or 3-level VE model (specified by the vePeriods vector with either 2 or 3 components, respectively, and aveVE). Under the option "half", both the 2- and 3-level VE model assumes a maximal VE in the second time interval such that, when halved in the first interval, the weighted average of VE over the first two time intervals equals aveVE. Only the first character is necessary.
aveVE	a numeric vector containing, for each treatment arm in N, a time-averaged vaccine efficacy (VE), defined as the weighted average of VEs in two or three time intervals specified by vePeriods. aveVE and vePeriods together characterize the VE model if VEmodel is set to "constant" or "half"; otherwise, aveVE is ignored. If VEmodel = "half", then aveVE is defined as follows: both the 2- and 3-level VE model assumes a maximal VE in the second time interval such that, when halved in the first interval, the weighted average of VE over the first two time intervals equals aveVE. aveVE is also applied thereafter. The components of N and aveVE correspond to each other.
vePeriods	a numeric vector defining start times (in weeks) of time intervals with (potentially) distinct VE levels depending on the choice of VEmodel. The value 1 in the vector signifies the beginning of follow-up. If VEmodel equals "constant", then vePeriods must have length 1 (typically then vePeriods <- 1). If VEmodel equals "half", then vePeriods must have length 2 or 3.
veByPeriod	a list (NULL by default) allowing to specify the VE level for each time interval in vePeriods for each treatment arm (including the control arm). The VE model can be specified in two ways: (a) by providing a vector of 'full' VE levels for the treatment arms starting with the control arm (component fullVE) and, for each treatment arm, vectors of fractions of fullVE pertaining to the time intervals (component C1 for the control arm, and T1, T2, etc. for the active arms), or (b) by providing, for each treatment arm, a vector of the actual VE levels pertaining to the time intervals (components C1, T1, T2, etc.). Note that the first component after fullVE, if it exists, must be C1 for the control arm, followed by the active arms. The vector fullVE, if it exists, has the same length as the number of arms, while the vectors C1, T1, T2, etc. have the same length as the number of time intervals in vePeriods. If the VE model is specified via veByPeriod, then aveVE will be ignored.
enrollPeriod	the final week of the enrollment period
enrollPartial	the final week of the portion of the enrollment period with a reduced enrollment rate defined by enrollPartialRelRate
enrollPartialRelRate	a non-negative value characterizing the fraction of the weekly enrollment rate governing enrollment from week 1 until week enrollPartial

dropoutRate	a (prior) dropout probability within 1 year
infecRate	a (prior) infection probability within 1 year in the control arm
fuTime	a follow-up time (in weeks) of each participant
visitSchedule	a numeric vector listing the visit weeks at which testing for the endpoint is conducted
missVaccProb	a numeric vector with conditional probabilities of having missed a vaccination given the follow-up time exceeds VEcutoffWeek weeks. For each component, a separate per-protocol indicator is generated. Each per-protocol cohort includes subjects with (i) a non-missing vaccination, and (ii) follow-up time exceeding VEcutoffWeek weeks. If NULL, no per-protocol indicators are included.
VEcutoffWeek	a time cut-off (in weeks); the follow-up time exceeding VEcutoffWeek weeks is required for inclusion in the per-protocol cohort
nTrials	the number of trials to be simulated
blockSize	a constant block size to be used in permuted-block randomization. The choice of blockSize requires caution to achieve the desired balance of treatment assignments within a block.
stage1	the final week of stage 1 in a two-stage trial
saveFile	a character string specifying the name of the output .RData file. If NULL (default), a default file name will be used.
saveDir	a character string specifying a path for the output directory. If supplied, the output is saved as an .RData file in the directory; otherwise the output is returned as a list.
verbose	a logical value indicating whether information on the output directory and file name should be printed out (default is TRUE)
randomSeed	sets seed of the random number generator for simulation reproducibility

Details

All time variables use week as the unit of time. Month is defined as 52/12 weeks.

The prior weekly enrollment rate is calculated based on the duration of the enrollment periods with reduced/full enrollment rates and the total number of subjects to be enrolled.

The weekly enrollment, dropout and infection rates used for generating trial data are sampled from specified prior distributions (the prior annual dropout and infection probabilities are specified by the user). The default choice considers non-random point-mass distributions, i.e., the prior rates directly govern the accumulation of trial data.

Subjects' enrollment is assumed to follow a Poisson process with a time-varying rate (the argument enrollPartialRelRate characterizes a reduced enrollment rate applied to weeks 1 through enrollPartial, i.e., full enrollment starts at week enrollPartial+1). The number of enrolled subjects is determined by the vector N.

Dropout times are assumed to follow an exponential distribution where the probability of a dropout within 1 week is equal to dropoutRate/52.

Permuted-block randomization is used for assigning treatment labels. If left unspecified by the user, an appropriate block size, no smaller than 10, will computed and used. The function getBlockSize can be used to determine appropriate block sizes (see help(getBlockSize)).

Infection times are generated following the VE schedule characterized by `aveVE`, `VEmodel` and `vePeriods`. Independent exponential times are generated within each time period of constant VE, and their minimum specifies the right-censored infection time. Exponential rates are chosen that satisfy the user-specified requirements on the treatment- and time-period-specific probabilities of an infection within 1 week (in the control arm, the infection probability within 1 week uniformly equals `infecRate`/52).

Infection diagnosis times are calculated according to the `visitSchedule`. The observed follow-up time is defined as the minumum of the infection diagnosis time, dropout time, and `fuTime`.

Value

If `saveDir` is specified, the output list (named `trialObj`) is saved as an `.RData` file (the output directory path is printed); otherwise it is returned. The output object is a list with the following components:

- `trialData`: a list with `nTrials` components each of which is a `data.frame` with at least the variables `trt`, `entry`, `exit`, and `event` storing the treatment assignments, enrollment times, study exit times, and event indicators, respectively. The observed follow-up times can be recovered as `exit - entry`. Indicators of belonging to the per-protocol cohort (named `pp1`, `pp2`, etc.) are included if `missVaccProb` is specified.
- `NinfStage1`: a list whose components are numeric vectors with the numbers of `stage1` infections by treatment (`[1]` = control arm) for each simulated trial
- `nTrials`: the number of simulated trials
- `N`: the total number of enrolled trial participants
- `nArms`: the number of treatment arms
- `trtAssgnProbs`: a numeric vector containing the treatment assignment probabilities
- `blockSize`: the block size used for treatment assignment
- `fuTime`: the follow-up time (in weeks) of each participant
- `rates`: a list with three components: the prior weekly enrollment rate (`enrollment`), the prior probability of dropout within 1 week (`dropout`), and the prior probability of infection within 1 week (`infection`)
- `enrollSchedule`: a `data.frame` summarizing information on enrollment periods and corresponding relative enrollment rates (relative to the weekly "base" enrollment rate). The column names are `start`, `end`, and `relativeRates`.
- `VEs`: a list with components being numeric vectors containing VE levels assumed within time periods defined by `vePeriods` for each active treatment arm
- `infecRates`: a `data.frame` summarizing information on time periods of distinct VE across all treatment arms. The variables `trt`, `start`, `end`, and `relRate` carry treatment assignment labels, first and last week of a time interval, and the pertaining assumed hazard ratio in the given interval.
- `randomSeed`: the set seed of the random number generator for simulation reproducibility

See Also

`monitorTrial`, `censTrial`, and `rankTrial`

Examples

```
### constant VE throughout the follow-up period
simData <- simTrial(N=c(1000, rep(700, 2)), VEmodel="constant",
                    aveVE=seq(0, 0.4, by=0.2), vePeriods=1,
                    enrollPeriod=78, enrollPartial=13, enrollPartialRelRate=0.5,
                    dropoutRate=0.05, infecRate=0.04, fuTime=156,
                    visitSchedule=seq(0, 156, by=4),
                    missVaccProb=c(0,0.05,0.1,0.15), VEcutoffWeek=26, nTrials=5,
                    stage1=78, randomSeed=300)

### a 3-level VE model specified by 'aveVE' and 'vePeriods'
simData <- simTrial(N=c(1000, rep(700, 2)), VEmodel="half",
                    aveVE=seq(0, 0.4, by=0.2), vePeriods=c(1, 27, 79),
                    enrollPeriod=78, enrollPartial=13, enrollPartialRelRate=0.5,
                    dropoutRate=0.05, infecRate=0.04, fuTime=156,
                    visitSchedule=seq(0, 156, by=4),
                    missVaccProb=c(0,0.05,0.1,0.15), VEcutoffWeek=26, nTrials=5,
                    stage1=78, randomSeed=300)

### a manually entered VE model specified by 'veByPeriod' using 'fullVE'
### and fractions
simData <- simTrial(N=c(1000, rep(700, 2)), VEmodel="manual",
                    vePeriods=c(1, 15, 27, 79),
                    veByPeriod=list(fullVE=c(0, 0.6, 0.8),
                                    C1=c(1, 1),
                                    T1=c(0.1, 2/3, 1, 0.75),
                                    T2=c(0.1, 0.5, 1, 0.9)),
                    enrollPeriod=78, enrollPartial=13, enrollPartialRelRate=0.5,
                    dropoutRate=0.05, infecRate=0.04, fuTime=156,
                    visitSchedule=seq(0, 156, by=4),
                    missVaccProb=c(0,0.05,0.1,0.15), VEcutoffWeek=26, nTrials=5,
                    stage1=78, randomSeed=300)

### the same manually entered VE model as above by specifying the actual
### VE levels in 'veByPeriod'
simData <- simTrial(N=c(1000, rep(700, 2)), VEmodel="manual",
                    vePeriods=c(1, 15, 27, 79),
                    veByPeriod=list(C1=c(0, 0, 0, 0),
                                    T1=c(0.1, 2/3, 1, 0.75) * 0.6,
                                    T2=c(0.1, 0.5, 1, 0.9) * 0.8),
                    enrollPeriod=78, enrollPartial=13, enrollPartialRelRate=0.5,
                    dropoutRate=0.05, infecRate=0.04, fuTime=156,
                    visitSchedule=seq(0, 156, by=4),
                    missVaccProb=c(0,0.05,0.1,0.15), VEcutoffWeek=26, nTrials=5,
                    stage1=78, randomSeed=300)

### alternatively, to save the .RData output file (no '<-' needed):
###
### simTrial(N=c(1400, rep(1000, 2)), aveVE=seq(0, 0.4, by=0.2), VEmodel="half",
###          vePeriods=c(1, 27, 79), enrollPeriod=78, enrollPartial=13,
###          enrollPartialRelRate=0.5, dropoutRate=0.05, infecRate=0.04, fuTime=156,
###          visitSchedule=seq(0, 156, by=4),
```

```
###          missVaccProb=c(0,0.05,0.1,0.15), VEcutoffWeek=26, nTrials=5,
###          stage1=78, saveDir="./", randomSeed=300)
```

`tabEventAccrual` *Tabulate event accrual over time since first enrollment*

Description

Tabulates side-by-side different event totals and time periods since first enrollment required to accrue the event totals. The user specifies a vector of either event totals or time points since first enrollment, and `tabEventAccrual` completes the table.

Usage

```
tabEventAccrual(
  trialData,
  atEvents = NULL,
  atWeeks = NULL,
  prob = 0.5,
  lagTimeMITT = 0,
  lagTimePP = NULL,
  namePP = "pp1",
  na.ub = 0.2
)
```

Arguments

`trialData`	either a list of data frames from `simTrial` (i.e., component `trialData` from the output list) or a character string specifying a path to an `.RData` file outputted by `simTrial`
`atEvents`	a numeric vector specifying treatment-pooled event counts for which empirical quantiles of time-to-accrual shall be calculated
`atWeeks`	a numeric vector specifying time points (in weeks) since first enrollment for which empirical quantiles of treatment-pooled event counts shall be calculated
`prob`	a numeric value in $(0, 1)$ specifying the probability at which the empirical quantiles across the simulated trials are computed (default is 0.5)
`lagTimeMITT`	a time point (in weeks). Only events with time-to-event $\geq$ `lagTimeMITT` are counted in the MITT column (default is 0).
`lagTimePP`	a time point (in weeks). If specified, only PP events with time-to-event $\geq$ `lagTimePP` are counted in the PP column.
`namePP`	a character string specifying the name of the column in each data frame in `trialData` which indicates membership in the PP cohort (default is "pp1")
`na.ub`	a numeric value specifying an upper limit on the fraction of simulated trials that do not reach a given event count in `atEvents` to still compute the empirical quantile of time-to-accrual. If the fraction of such trials exceeds `na.ub`, `NA` will be produced.

Details

All time variables use week as the unit of time.

If the user specifies `atEvents`, time periods since first enrollment are computed that are needed to observe `atEvents` MITT and `atEvents` PP events.

If the user specifies `atWeeks`, MITT and PP event totals are computed that are observed by `atWeeks` weeks since first enrollment.

The function inputs a large number of simulated trial data, and the computed variables (time periods or event totals) are empirical quantiles at probability `prob` of the sample distributions (of time periods or event totals). Medians are computed by default.

Value

A data frame (with at least two columns) of event totals and associated time periods since first enrollment required to accrue the event totals in the MITT cohort. If an PP cohort is specified (via `lagTimePP`), a third column is added.

See Also

`simTrial`

Examples

```
simData <- simTrial(N=c(1000, rep(700, 2)), aveVE=seq(0, 0.4, by=0.2),
                    VEmodel="half", vePeriods=c(1, 27, 79), enrollPeriod=78,
                    enrollPartial=13, enrollPartialRelRate=0.5, dropoutRate=0.05,
                    infecRate=0.04, fuTime=156,
                    visitSchedule=c(0, (13/3)*(1:4), seq(13*6/3, 156, by=13*2/3)),
                    missVaccProb=c(0,0.05,0.1,0.15), VEcutoffWeek=26, nTrials=5,
                    blockSize=NULL, stage1=78, randomSeed=300)

## user specifies MITT event totals
tabEventAccrual(simData$trialData, atEvents=seq(10, 100, by=10))

## user specifies MITT and PP event totals
tabEventAccrual(simData$trialData, atEvents=seq(10, 100, by=10), lagTimePP=6)

## user specifies time points since first enrollment
tabEventAccrual(simData$trialData, atWeeks=seq(52, 156, by=8), lagTimePP=6)
```

VEpowerPP — *Unconditional Power to Detect Positive Treatment Efficacy in a Per-Protocol Cohort*

Description

`VEpowerPP` computes unconditional power to detect positive treatment (vaccine) efficacy in per-protocol cohorts identified in `simTrial`-generated data-sets.

Usage

```
VEpowerPP(
  dataList,
  lowerVEuncPower,
  alphaUncPower,
  VEcutoffWeek,
  stage1,
  outName = NULL,
  saveDir = NULL,
  verbose = TRUE
)
```

Arguments

dataList	if saveDir = NULL, a list of objects (lists) returned by censTrial; otherwise a list of .RData file names (character strings) generated by censTrial
lowerVEuncPower	a numeric value specifying a one-sided null hypothesis H0: VE(VEcutoffWeek–stage1) $\leq$ lowerVEuncPower x 100%. Unconditional power (i.e., accounting for sequential monitoring) to reject H0 in the per-protocol cohort is calculated, where the rejection region is defined by the lower bound of the two-sided (1-alphaUncPower) x 100% confidence interval for VE(VEcutoffWeek–stage1) being above lowerVEuncPower (typically a number in the 0–0.5 range).
alphaUncPower	one minus the nominal confidence level of the two-sided confidence interval used to test the one-sided null hypothesis H0: VE(VEcutoffWeek–stage1) $\leq$ lowerVEuncPower x 100% against the alternative hypothesis H1: VE(VEcutoffWeek–stage1) $>$ lowerVEuncPower x 100%.
VEcutoffWeek	a cut-off time (in weeks). Only subjects with the follow-up time exceeding VEcutoffWeek are included in the per-protocol cohort.
stage1	the final week of stage 1 in a two-stage trial
outName	a character string specifying the output .RData file name. If outName = NULL but saveDir is specified, the output file is named VEpwPP.RData.
saveDir	a character string specifying a path for the output directory. If supplied, the output is saved as an .RData file named outName in the directory; otherwise the output is returned as a list.
verbose	a logical value indicating whether information on the output directory and file name should be printed out (default is TRUE)

Details

All time variables use week as the unit of time. Month is defined as 52/12 weeks.

A per-protocol cohort indicator is assumed to be included in the simTrial-generated data-sets, which is ensured by specifying the missVaccProb argument in simTrial.

VE(VEcutoffWeek–stage1) is estimated as one minus the ratio of Nelson-Aalen-based cumulative incidence functions. VEpowerPP computes power to reject the null hypothesis H0: VE(VEcutoffWeek–stage1) $\leq$ lowerVEuncPower x 100%. H0 is rejected if the lower bound of the two-sided (1-alphaUncPower) x 100% confidence interval for VE(VEcutoffWeek–stage1) lies above lowerVEuncPower.

Value

If saveDir is specified, the output list (named pwList) is saved as an .RData file named outName (or VEpwPP.RData if left unspecified); otherwise the output list is returned. The output object is a list (of equal length as dataList) of lists with the following components:

- VE: a numeric vector of VE(VEcutoffWeek–stage1) estimates for each missing vaccination probability in missVaccProb of simTrial
- VEpwPP: a numeric vector of powers to reject the null hypothesis H0: VE(VEcutoffWeek–stage1) $\leq$ lowerVEuncPower x 100% for each missing vaccination probability in missVaccProb of simTrial

See Also

simTrial

Examples

```
simData <- simTrial(N=rep(1000, 2), aveVE=c(0, 0.4), VEmodel="half",
                    vePeriods=c(1, 27, 79), enrollPeriod=78,
                    enrollPartial=13, enrollPartialRelRate=0.5, dropoutRate=0.05,
                    infecRate=0.04, fuTime=156,
                    visitSchedule=c(0, (13/3)*(1:4), seq(13*6/3, 156, by=13*2/3)),
                    missVaccProb=c(0,0.05,0.1,0.15), VEcutoffWeek=26, nTrials=5,
                    stage1=78, randomSeed=300)

monitorData <- monitorTrial(dataFile=simData, stage1=78, stage2=156,
                            harmMonitorRange=c(10,100), alphaPerTest=NULL,
                            nonEffStartMethod="FKG", nonEffInterval=20,
                            lowerVEnoneff=0, upperVEnoneff=0.4,
                            highVE=0.7, stage1VE=0, lowerVEuncPower=0,
                            alphaNoneff=0.05, alphaHigh=0.05, alphaStage1=0.05,
                            alphaUncPower=0.05, estimand="cuminc", lagTime=26)

censData <- censTrial(dataFile=simData, monitorFile=monitorData, stage1=78, stage2=156)

VEpwPP <- VEpowerPP(dataList=list(censData), lowerVEuncPower=0, alphaUncPower=0.05,
                    VEcutoffWeek=26, stage1=78)

### alternatively, to save the .RData output file (no '<-' needed):
###
### simTrial(N=rep(1000, 2), aveVE=c(0, 0.4), VEmodel="half",
###          vePeriods=c(1, 27, 79), enrollPeriod=78, enrollPartial=13,
###          enrollPartialRelRate=0.5, dropoutRate=0.05, infecRate=0.04, fuTime=156,
###          visitSchedule=c(0, (13/3)*(1:4), seq(13*6/3, 156, by=13*2/3)),
###          missVaccProb=c(0,0.05,0.1,0.15), VEcutoffWeek=26, nTrials=5,
###          stage1=78, saveDir="./", randomSeed=300)
###
### monitorTrial(dataFile=
###          "simTrial_nPlac=1000_nVacc=1000_aveVE=0.4_infRate=0.04.RData",
###          stage1=78, stage2=156, harmMonitorRange=c(10,100), alphaPerTest=NULL,
###          nonEffStartMethod="FKG", nonEffInterval=20,
```

```
###          lowerVEnoneff=0, upperVEnoneff=0.4, highVE=0.7, stage1VE=0,
###          lowerVEuncPower=0, alphaNoneff=0.05, alphaHigh=0.05, alphaStage1=0.05,
###          alphaUncPower=0.05, estimand="cuminc", lagTime=26, saveDir="./")
###
### censTrial(dataFile=
###  "simTrial_nPlac=1000_nVacc=1000_aveVE=0.4_infRate=0.04.RData",
###  monitorFile=
###  "monitorTrial_nPlac=1000_nVacc=1000_aveVE=0.4_infRate=0.04_cuminc.RData",
###  stage1=78, stage2=156, saveDir="./")
###
### VEpowerPP(dataList=
###  list("trialDataCens_nPlac=1000_nVacc=1000_aveVE=0.4_infRate=0.04_cuminc.RData"),
###  lowerVEuncPower=0, alphaUncPower=0.05, VEcutoffWeek=26, stage1=78, saveDir="./")
```

Index