Package ‘sensitivityPStrat’

February 20, 2015

Type Package

Title Principal Stratification Sensitivity Analysis Functions

Version 1.0-6

Date 2014-11-04

Author Charles Dupont, Bryan Shepherd

Maintainer Charles Dupont <charles.dupont@vanderbilt.edu>

Description
This package provides functions to perform principal stratification sensitivity analyses on datasets.

Imports stats, survival

License GPL

LazyLoad yes

NeedsCompilation yes

Repository CRAN

Date/Publication 2014-12-04 00:14:37

R topics documented:

sensitivityPStrat-package *Principal Stratification Sensitivity Analysis Functions*

Description

This package provides functions to perform sensitivity analyses of treatment effects within principal strata.

Details

A treatment effect is a contrast between $Y(0)$ and $Y(1)$ where $Y(0)$ is the outcome if not treated and $Y(1)$ is the outcome if treated. The average treatment effect (or average causal effect) is $E(Y(1) - Y(0))$. In some settings there may be interest in estimating the average treatment effect among those who would be selected under either treatment assignment (i.e., $E(Y(1) - Y(0)|S(0) = S(1) = 1)$, where $S(0)$ is the indicator of selection if not treated and $S(1)$ is the indicator of selection if treated (Robins 1986). For example, one may want to assess the average treatment effect of a drug on quality of life among those who would have lived regardless of their treatment assignment. The subgroup defined by $S(0) = S(1) = 1$ (e.g., those who would have lived regardless of treatment assignment) has been referred to as a principal stratum (Frangakis and Rubin, 2002). Principal stratum membership is not known so to identify the average treatment effect (or related estimands) within a principal stratum we assume 1. SUTVA (Rubin 1978) (i.e., no interference – that the potential outcomes for all subjects are independent of the treatment assignment of other subjects),2. ignorable treatment assignment (i.e., random assignment of treatment), 3. that one of the principal strata is empty, and 4. that a selected subject's outcome if assigned one treatment is independent of selection if assigned the other treatment. This package implements sensitivity analysis methods that relax these latter two assumptions.

`sensitivityHHS` and `sensitivityGBH` implement the methods described by Hudgens, Hoering and Self (2003) and Gilbert, Bosch, and Hudgens (2003), respectively. They estimate the average treatment effect in the always-selected principal stratum under assumptions 1-3, relaxing 4 using a worse-case scenario analysis (`sensitivityHHS`) or using a sensitivity parameter (`sensitivityGBH`). These functions also have options to do rank-based analyses and to compute other measures of treatment efficacy with continuous or binary outcomes (Hudgens and Halloran, 2006). `sensitivitySGL` implements the methods described by Shepherd, Gilbert, and Lumley (2006). It is similar to `sensitivityHHS` and `sensitivityGBH` except that it computes the difference between distribution functions in the always-selected principal stratum and allows the outcome to be right-censored. `sensitivityJR` estimates the average treatment effect in the always-selected principal stratum relaxing assumptions 3 and 4 as described by Jemiai and Rotnitzky (2005) and Shepherd, Redman, and Ankerst (2008). `sensitivitySGD` incorporates the methods of Shepherd, Gilbert, and Dupont (in press), extending `sensitivityJR` to right-censored outcomes.

Author(s)

Bryan E. Shepherd
Department of Biostatistics
Vanderbilt University

Charles Dupont
Department of Biostatistics
Vanderbilt University
<charles.dupont@vanderbilt.edu>

Maintainer: Charles Dupont

References

Frangakis CE and Rubin DB (2002), "Principal stratification in causal inference," Biometrics 58, 21-29.

Gilbert PB, Bosch RJ, and Hudgens MG (2003), "Sensitivity Analysis for the Assessment of Causal Vaccine Effects of Viral Load in HIV Vaccine Trials," Biometrics 59, 531-541.

Hudgens MG, Halloran ME, "Causal vaccine effects on binary post infection outcomes," Journal of the American Statisitcal Association 101, 51-64.

Hudgens MG, Hoering A, and Self SG (2003), "On the Analysis of Viral Load Endpoints in HIV Vaccine Trials," Statistics in Medicine 22, 2281-2298.

Jemiai Y (2005), "Semiparametric Methods for Inferring Treatment Effects on Outcomes Defined Only if a Post-Randomization Event Occurs," unpublished doctoral dissertation under the supervision of A. Rotnitzky, Harvard School of Public Health, Dept. of Biostatistics.

Robins JM (1986), "A new approach to causal inference in mortality studies with sustained exposure periods - Application to control of the healthy worker survivor effect," Mathematical Modeling 7, 1393-1512.

Rubin DB (1978), "Bayesian inference for causal effects: the role of randomization," The Annals of Statistics 6, 34-58.

Shepherd BE, Gilbert PB, Lumley T (2007), "Sensitivity analyses comparing time-to-event outcomes existing only in a subset selected postrandomization," Journal of the American Statistical Association 102, 573-582.

Shepherd BE, Gilbert PB, and Dupont CT, "Sensitivity analyses comparing time-to-event outcomes only existing in a subset selected postrandomization and relaxing monotonicity," Biometrics (in press).

See Also

`Surv`

`calc.v` *Calculates the v matrix used in the estimation of standard errors in sensitivitySGL.*

Description

Calculates the v matrix used in the estimation of standard errors in sensitivitySGL.

Usage

```
calc.v(event, time)
```

Arguments

`event`	logical vector indicating whether and event has happened.
`time`	vector; time until event or observation halted.

Value

returns a matrix.

Author(s)

Bryan E. Shepherd
Department of Biostatistics
Vanderbilt University

Charles Dupont
Department of Biostatistics
Vanderbilt University

References

Shepherd BE, Gilbert PB, Lumley T (2007), "Sensitivity analyses comparing time-to-event outcomes existing only in a subset selected postrandomization," Journal of the American Statistical Association 102, 573-582.

funArray *Create an array of functions*

Description

Creates a array of functions.

Usage

```
funArray(...)
```

Arguments

`...`	passed to `array`. see arguments to `array`

Author(s)

Charles Dupont
Department of Biostatistics
Vanderbilt University

See Also

`funVector`, `funMatrix`, `array`

`funMatrix` *Create a matrix of functions*

Description

Creates a matrix of functions.

Usage

```
funMatrix(...)
```

Arguments

`...`	passed to `matrix`. see arguments to `matrix`

Author(s)

Charles Dupont
Department of Biostatistics
Vanderbilt University

See Also

`funVector`, `funArray`, `matrix`

funVector *Create a vector of functions*

Description

Creates a vector of functions.

Usage

```
funVector(length = 0)
```

Arguments

`length`	integer; length of vector.

Author(s)

Charles Dupont
Department of Biostatistics
Vanderbilt University

See Also

`funMatrix`, `funArray`, `vector`

plot.sensitivity *plots the results of calls to the sensitivity functions.*

Description

Functions used to plot the objects created by the sensitivityPStrat family of functions.

Usage

```
## S3 method for class 'sensitivity.1.0d'
plot(x, xlim, ylim,
     xlab = expression(beta), ylab = "ACE",
     display = c("analytic", "bootstrap"),
     ci.select = 1,
     col = "black", line.col = col, point.col = col,
     analytic.col = "red", analytic.line.col = analytic.col,
     analytic.point.col = analytic.col,
     bootstrap.col = "green", bootstrap.line.col = bootstrap.col,
     bootstrap.point.col = bootstrap.col,
     panel.last = NULL, type = "l", ...)
```

```
## S3 method for class 'sensitivity.2.0d'
plot(x, xlim, ylim, xlab = expression(beta[0]), ylab = expression(beta[1]),
    display = c("analytic", "bootstrap"), col = c(gray(.9), gray(1), gray(.8)),
     panel.last = NULL, ...)

## S3 method for class 'sensitivity.1.1d'
plot(x, xlim, ylim,
     xlab = expression(beta), ylab = "SCE",
     t.point, display = c("analytic", "bootstrap"),
     col = "black", line.col = col, point.col = col,
     analytic.col = "red", analytic.line.col = analytic.col,
     analytic.point.col = analytic.col,
     bootstrap.col = "green", bootstrap.line.col = bootstrap.col,
     bootstrap.point.col = bootstrap.col,
     panel.last = NULL, type = "l", ...)
```

Arguments

`x`	sensitivity object
`t.point`	the time point at which data to create the plot.
`display`	character vector. Controls which confidence interval to use plot.
`ci.select`	integer vector or ‘`all`’. Selects the confidence interval to be ploted. If set to ‘`all`’ then all confidence intervals are plotted. Default value is 1.
`line.col`	the color all the lines should be.
`point.col`	the color all the infinity points should be.
`analytic.col`	vector; the color of all of the analytic confidence interval markings. Value are recycled if more confidence intervals are selected then given color values.
`analytic.line.col`	vector;the color of all of the analytic confidence interval lines. Value are recycled if more confidence intervals are selected then given color values.
`analytic.point.col`	vector; the color of all of the analytic confidence interval infinity points. Value are recycled if more confidence intervals are selected then given color values.
`bootstrap.col`	vector; the color of all of the bootstrap confidence interval markings. Value are recycled if more confidence intervals are selected then given color values.
`bootstrap.line.col`	vector; the color of all of the bootstrap confidence interval lines. Value are recycled if more confidence intervals are selected then given color values.
`bootstrap.point.col`	vector; the color of all of the bootstrap confidence interval infinity points. Value are recycled if more confidence intervals are selected then given color values.
`xlim, ylim, xlab, ylab, col, panel.last, type`	see `plot.default`
`...`	arguments passed to `plot.default`

Author(s)

Charles Dupont
Department of Biostatistics
Vanderbilt University

See Also

plot.default

Examples

```
data(vaccine.trial)

ansJR<-with(vaccine.trial,
          sensitivityJR(z=treatment,s=hiv.outcome,y=logVL,
                    beta0=c(-1,-.5,-.25,0,.25,.5,1),
                    beta1=c(-1,-.5,-.25,0,.25,.5,1),
                    phi=c(0.95,0.90), selection="infected",
                    groupings=c("placebo","vaccine"),
                    N.boot=50)
         )

plot(ansJR)

ans<-with(vaccine.trial,
          sensitivityGBH(z=treatment,s=hiv.outcome,y=logVL,
                    beta=c(-Inf,-1,0.75,-0.5,-0.25,0,.25,.5,.75,1,Inf),
                    selection="infected",
                    groupings=c("placebo","vaccine"),
                    empty.principal.stratum=c("not infected","infected"),
                    ci.method="bootstrap", ci=c(0.95, 0.9, 0.9),
                    ci.type=c("twoSided", "upper", "lower"),
                    custom.FUN=function(mu0, mu1, ...) mu1 - mu0,
                    N.boot=50, method=c("ACE", "T1", "T2"),
                    upperTest=TRUE, lowerTest=TRUE, twoSidedTest=TRUE)
         )

plot(ans, ci.select="all", bootstrap.col=c("red","green","blue"))
```

print.sensitivity *prints the results of calls to the sensitivity functions.*

Description

Print the prints sensitivityPStrat objects in a visually understandable way.

Usage

```
## S3 method for class 'sensitivity.0d'
print(x, ...)
## S3 method for class 'sensitivity.1d'
print(x, ...)
```

Arguments

`x`	sensitivity object
`...`	arguments passed to other print methods

Author(s)

Charles Dupont
Department of Biostatistics
Vanderbilt University

See Also

`print.default`

Examples

```
data(vaccine.trial)

print(with(vaccine.trial,
          sensitivityJR(z=treatment,s=hiv.outcome,y=logVL,
                    beta0=c(-1,-.75,-.5,-.25,0,.25,.5,.75,1),
                    beta1=c(-1,-.75,-.5,-.25,0,.25,.5,.75,1),
                    phi=c(0.95,0.90), selection="infected",
                    groupings=c("placebo","vaccine"),
                    N.boot=50)
         ))
```

`sensitivityGBH` *Principal stratification sensitivity analysis.*

Description

Performs a sensitivity analysis using the method described in Gilbert, Bosch, and Hudgens (2003).

Usage

```
sensitivityGBH(z, s, y, beta, selection, groupings,
               empty.principal.stratum, ci = 0.95,
               ci.method = c("analytic", "bootstrap"),
               ci.type = "twoSided", custom.FUN = NULL, na.rm = FALSE,
               N.boot = 100, interval = c(-100, 100),
               upperTest = FALSE, lowerTest = FALSE, twoSidedTest = TRUE,
               method = c("ACE", "T1", "T2"), isSlaveMode=FALSE)
```

Arguments

`z` vector; contains the grouping values (e.g., treatment assignment) for each record.

`s` vector; indicates whether a record is selected.

`y` vector; outcome value. Can be `NA` for unselected records.

`beta` vector; values of the β sensitivity parameter. `Inf` and `-Inf` are acceptable.

`selection` The value of `s` indicating selection.

`groupings` vector of two elements `c(`*g0*`,`*g1*`)`; describes the possible group values. The first element *g0* being the value of `z` that delineates the first group, the last element *g1* being the value of `z` that delineates the second group.

`empty.principal.stratum`
vector of two elements `c(`*s0*`,`*s1*`)`; describes the `s` values that select the empty principal stratum. If `empty.principal.stratum=c(`*s0*`,`*s1*`)`, then stratum defined by $S(g0) = s0$ and $S(g1) = s1$ is the empty stratum. In this example *s0* and *s1* refer to the two possible values of `s`. (Note: method only works if $s0 \neq s1$).

`ci` numeric vector; confidence interval level. Defaults to `0.95`

`ci.method` character; method by which the confidence interval and variance are calculated. Can be “analytic” or “bootstrap”. Defaults to `c("analytic","bootstrap")`

`ci.type` character vector; type of confidence interval that the corresponding `ci` element is referring to. Can be “upper”, “lower”, or “twoSided”. Defaults to `"twoSided"`.

`custom.FUN` function; function to calculate custom result. `mu0`, `mu1`, `p0`, `p1` are available to be used as arguments in the custom function, where `mu0` $= E(Y(g0)|S(g0) = S(g1) = selected)$, `mu1` $= E(Y(g1)|S(g0) = S(g1) = selected)$, `p0` $= P(S(g0) = selected)$, and `p1` $= P(S(g1) = selected)$. The custom function must return a single value.

`na.rm` logical; indicates whether records that are invalid due to `NA` values should be removed from the data set.

`N.boot` integer; number of bootstrap repetitions that will be run when `ci.method` includes “bootstrap”.

`interval` numeric vector of length 2. Controls the range limits used by `optimize` to estimate α.

`lowerTest` logical. Return the lower one sided p-value for returned tests. Defaults to `FALSE`

`upperTest` logical. Return the upper one sided p-value for returned tests. Defaults to `FALSE` `twoSidedTest` logical. Return a two sided p-value for returned tests. Defaults to `TRUE`

`method` character vector; type of test statistic calculated. Can be one or more of “ACE”, “T1”, or “T2”. Defaults to `"ACE"`. Methods “T1” and “T2” are not implemented if `ci.method` includes “analytic”.

`isSlaveMode` logical. Internal Use only. Used in recursion.

Details

Performs a sensitivity analysis estimating the average causal effect among those who would have been selected regardless of treatment assignment (ACE). The method assumes no interference (i.e., potential outcomes of all subjects are unaffected by treatment assignment of other subjects), ignorable (i.e., random) treatment assignment, and monotonicity (i.e., one of the principal strata is empty). ACE is identified by assuming a value of the sensitivity parameter beta, where e^{β} has an odds ratio interpretation:

If `empty.principal.stratum` $= c(S(g0) = \mathit{not\ selected}, S(g1) = \mathit{selected})$ then given selected if assigned *g0*, the odds of being selected if assigned *g1* multiplicatively increase e^{β} for every 1-unit increase in $Y(g0)$.

If `empty.principal.stratum` $= c(S(g0) = \mathit{selected}, S(g1) = \mathit{not\ selected})$ then given selected if assigned *g1*, the odds of being selected if assigned *g0* multiplicatively increase e^{β} for every 1-unit increase in $Y(g1)$.

Specifying `beta=-Inf` or `beta=Inf` calls `sensitivityHHS`.

T1 and T2 are rank-based analogs of ACE. See <REF TBD>.

Value

an object of class `sensitivity2d`.

`ACE` vector; $ACE = E(Y(g1) - Y(g0)|S(g1) = S(g0) = \texttt{selection})$. Vector of the estimated ACE values for specified `beta` values. Only exists if `method` includes “ACE”.

`ACE.ci` array; confidence interval of ACE determined by quantiles of bootstrap if `ci.method` includes “bootstrap”. Otherwise calculated using analytic variance with large sample normal approximation. Only exists if `method` includes “ACE”.

`ACE.var` vector; estimated variance of ACE. Only exists if `method` includes “ACE”.

`ACE.p` vector; estimated p-value of ACE. Only exists if `method` includes “ACE”.

`T1` vector; Vector of the estimated T1 test statistic for specified `beta` values. Only exists if `method` includes “T1”.

`T1.p` vector; estimated p-value of T1. Only exists if `method` includes “T1”.

`T2` vector; Vector of the estimated T2 statistic for specified `beta` values. Only exists if `method` includes “T2”.

`T2.p` vector; estimated p-value of T2. Only exists if `method` includes “T2”.

`beta` vector; user-specified β values

`alphahat` vector; estimated values of α

`Fas0` function; estimator for the empirical distribution function values for *y0* in the first group in the always selected principal stratum. $Pr(Y(g0) \leq y0|S(g0) = S(g1) = \texttt{selection}; \beta)$

`Fas1` function; estimator for the empirical distribution function values for *y1* in the second group in the always selected principal stratum. $Pr(Y(g1) \leq y1|S(g0) = S(g1) = \texttt{selection}; \beta)$

Author(s)

Bryan E. Shepherd
Department of Biostatistics
Vanderbilt University

Charles Dupont
Department of Biostatistics
Vanderbilt University

References

Gilbert PB, Bosch RJ, and Hudgens MG (2003), "Sensitivity Analysis for the Assessment of Causal Vaccine Effects of Viral Load in HIV Vaccine Trials," Biometrics 59, 531-541.

See Also

`sensitivityHHS`, `sensitivityJR`, `sensitivitySGL`

Examples

```
data(vaccine.trial)
ans<-with(vaccine.trial,
          sensitivityGBH(z=treatment,s=hiv.outcome,y=logVL,
                    beta=c(0,.25,.5,.75,1,1.25,1.5),
                    selection="infected",
                    groupings=c("placebo","vaccine"),
                    empty.principal.stratum=c("not infected","infected"),
                    N.boot=100)
         )
ans

ans<-with(vaccine.trial,
          sensitivityGBH(z=treatment,s=hiv.outcome,y=logVL,
                    beta=c(-Inf,-1,-0.75,-0.5,-0.25,0,.25,.5,.75,1,Inf),
                    selection="infected",
                    groupings=c("placebo","vaccine"),
                    empty.principal.stratum=c("not infected","infected"),
                    ci.method="bootstrap", ci=c(0.95, 0.9, 0.9),
                    ci.type=c('twoSided', 'upper', 'lower'),
                    custom.FUN=function(mu0, mu1, ...) mu1 - mu0,
                    N.boot=100, method=c("ACE", "T1", "T2"),
                    upperTest=TRUE, lowerTest=TRUE, twoSidedTest=TRUE)
          )
ans
```

`sensitivityHHS` *principal stratification sensitivity analysis using the HHS method.*

Description

Performs a principal stratification sensitivity analysis using the method described in Hudgens, Hoering, and Self (2003).

Usage

```
sensitivityHHS(z, s, y, bound = c("upper", "lower"), selection,
               groupings, empty.principal.stratum, ci = 0.95,
               ci.method = c("bootstrap", "analytic"),
               ci.type = "twoSided", custom.FUN = NULL, na.rm = FALSE,
               N.boot = 100, upperTest = FALSE, lowerTest = FALSE,
               twoSidedTest = TRUE, method = c("ACE", "T1", "T2"),
               isSlaveMode=FALSE)
```

Arguments

`z` vector; contains the grouping values (e.g., treatment assignment) for each record.

`s` vector; indicates whether a record is selected.

`y` vector; outcome values. Can be `NA` for unselected records.

`bound` vector; which bound should be calculated, “upper” and/or “lower”. Partial string matching is performed.

`selection` The value of `s` indicating selection.

`groupings` vector of two elements `c(`*g0*`,`*g1*`)`; describes to possible group values. The first element *g0* being the value of `z` which delineates the first group, the last element *g1* being the value of `z` which delineates the second group.

`empty.principal.stratum`
vector of two elements `c(`*s0*`,`*s1*`)`; describes the `s` values that select the empty principal stratum. If `empty.principal.stratum=c(`*s0*`,`*s1*`)`, then stratum defined by $S(g0) = s0$ and $S(g1) = s1$ is the empty stratum. In this example *s0* and *s1* refer to the two possible values of `s`. (Note: method only works if $s0 \neq s1$).

`ci` numeric vector; confidence interval level, defaults to `0.95`.

`ci.method` character; method by which the confidence interval and variance are calculated. Can be “analytic” or “bootstrap”. Defaults to `c("analytic","bootstrap")`. Currently only works for “bootstrap”.

`ci.type` character vector; type of confidence interval that the corisponding `ci` element is referring to. Can be “upper”, “lower”, or “twoSided”. Defaults to `"twoSided"`.

`custom.FUN` function; function to calculate custom result. `mu0`, `mu1`, `p0`, `p1` are available to be used as arguments in the custom function, where `mu0` $= E(Y(g0)|S(g0) = S(g1) = selected)$, `mu1` $= E(Y(g1)|S(g0) = S(g1) = selected)$, `p0` $= P(S(g0) = selected)$, and `p1` $= P(S(g1) = selected)$. The custom function must return a single value.

`na.rm` logical; indicates whether records that are invalid due to `NA` values should be removed from the data set.

`N.boot` integer. Number of bootstrap repetitions that will be run when `ci.method` includes "bootstrap".

`lowerTest` logical. Return the lower one sided p-value for returned tests. Defaults to `FALSE`

`upperTest` logical. Return the upper one sided p-value for returned tests. Defaults to `FALSE`

`twoSidedTest` logical. Return a two sided p-value for returned tests. Defaults to `TRUE`

`method` character vector; type of test statistic calculated. Can be one or more of "ACE", "T1", or "T2". Defaults to `"ACE"`.

`isSlaveMode` logical; Internal Use only. Used in recursion.

Details

Performs a sensitivity analysis estimating the average causal effect among those who would have been selected regardless of treatment assignment (ACE). The method assumes no interference (i.e., potential outcomes of all subjects are unaffected by treatment assignment of other subjects), ignorable (i.e., random) treatment assignment, and monotonicity (i.e., one of the principal strata is empty). ACE is still not identified after making these assumptions, so this method computes the lower and upper bounds of the estimated ACE. These bounds correspond to the values one would get if using `sensitivityGBH` and specifying the sensitivity parameter beta as `-Inf` or `Inf`.

Value

an object of class `sensitivity2d`.

`ACE` $ACE = E(Y(g1) - Y(g0)|S(g1) = S(g0) = \texttt{selection})$. Vector of the estimated ACE values at the specified bounds. Only exists if `method` includes "ACE".

`ACE.ci` vector; confidence interval of ACE determined by quantiles of bootstrap if `ci.method` includes "bootstrap". Otherwise calculated using analytic variance with large sample normal approximation (NOT YET WORKING). Only exists if `method` includes "ACE".

`ACE.var` vector; estimated variance of ACE. Only exists if `method` includes "ACE".

`ACE.p` vector; estimated p-value of ACE. Only exists if `method` includes "ACE".

`Fas0` function; estimator for the empirical distribution function values for *y0* in the first group in the always selected principal stratum at the bounds. $Pr(Y(g0) \leq y0|S(g0) = S(g1) = \texttt{selection})$

`Fas1` function; estimator for the empirical distribution function values for *y1* in the second group in the always selected principal stratum at the bounds. $Pr(Y(g1) \leq y1|S(g0) = S(g1) = \texttt{selection})$

Author(s)

Bryan E. Shepherd
Department of Biostatistics
Vanderbilt University

Charles Dupont
Department of Biostatistics
Vanderbilt University

References

Hudgens MG, Hoering A, and Self SG (2003), “On the Analysis of Viral Load Endpoints in HIV Vaccine Trials,” Statistics in Medicine 22, 2281-2298.

See Also

`sensitivityGBH`, `sensitivityJR`, `sensitivitySGL`

Examples

```
data(vaccine.trial)
est.bounds<-with(vaccine.trial,
                 sensitivityHHS(z=treatment, s=hiv.outcome, y=logVL,
                     selection="infected", groupings=c("placebo","vaccine"),
                     empty.principal.stratum=c("not infected","infected"),
                     N.boot=100)
                )
est.bounds

est.bounds<-with(vaccine.trial,
                 sensitivityHHS(z=treatment, s=hiv.outcome, y=logVL,
                     selection="infected", groupings=c("placebo","vaccine"),
                     empty.principal.stratum=c("not infected","infected"),
                     method=c("ACE", "T1", "T2"), N.boot=100,
                     custom.FUN=function(mu0, mu1, ...) mu1 - mu0,
                     upperTest=TRUE, lowerTest=TRUE, twoSidedTest=TRUE)
                )
est.bounds
```

`sensitivityJR` *Principal stratification sensitivity analysis relaxing the monotonicity assumption.*

Description

Principal stratification sensitivity analysis relaxing monotonicity as described by Jemiai and Rotnitzky (2005) and implemented by Shepherd, Redman, and Ankerst (2008).

Usage

```
sensitivityJR(z, s, y, beta0, beta1, phi, Pi, psi,
              selection, groupings,
              ci = 0.95, ci.method = c("analytic","bootstrap"),
              ci.type = "twoSided", custom.FUN=NULL, na.rm = FALSE,
              N.boot = 100, interval = c(-100, 100),
              upperTest = FALSE, lowerTest = FALSE, twoSidedTest = TRUE,
              verbose=getOption("verbose"), isSlaveMode = FALSE)
```

Arguments

`z` vector; contains the grouping values (e.g., treatment assignment) for each record.

`s` vector; indicates whether a record is selected.

`y` vector; outcome values. Can be `NA` for unselected records.

`beta0` vector; values of the sensitivity parameter $\beta 0$ linking outcome in group *g0* with selection if assigned group *g1*.

`beta1` vector; values of the sensitivity parameter $\beta 1$ linking outcome in group *g1* with selection if assigned group *g0*.

`phi, Pi, psi` vector; sensitivity parameters specifying the joint distribution of $S(g0)$, $S(g1)$. Only one of the three parameters should be specified. `psi` is the log-odds ratio of selection. `Pi` is the probability of being in the always selected principal stratum ($Pr(S(g0) = S(g1) = selected)$). `phi` is the probability of selection in group *g0* given selection in group *g1* ($Pr(S(g0) = 1|S(g1) = 1)$).

`selection` The value of `s` indicating selection.

`groupings` vector of two elements `c(`*g0*`,`*g1*`)`; describes to possible group values. The first element *g0* being the value of `z` the delineates the first group, the last element *g1* being the value of `z` which delineates the second group.

`ci` numeric vector; confidence interval value. Defaults to `0.95`

`ci.method` character; method by which the confidence interval and variance are calculated. Can be “analytic” or “bootstrap”. Defaults to `c("analytic","bootstrap")`

`ci.type` character vector; type of confidence interval that the corresponding `ci` element is referring to. Can be “upper”, “lower”, or “twoSided”. Defaults to `"twoSided"`.

`custom.FUN` function; function to calculate custom result. `mu0`, `mu1`, `p0`, `p1` are available to be used as arguments in the custom function, where `mu0` $= E(Y(g0)|S(g0) = S(g1) = selected)$, `mu1` $= E(Y(g1)|S(g0) = S(g1) = selected)$, `p0` $= P(S(g0) = selected)$, and `p1` $= P(S(g1) = selected)$. The custom function must return a single value.

`na.rm` logical; indicates whether records that are invalid due to `NA` values should be removed from the data set.

`N.boot` integer; number of bootstrap repetitions that will be run when `ci.method` includes “bootstrap”.

`interval` numeric vector of length 2. Controls the range limits used by optimize to estimate $\alpha 0$ and $\alpha 1$.

`lowerTest` logical. Return the lower one sided p-value for the ACE. Defaults to `FALSE` `upperTest` logical. Return the upper one sided p-value for the ACE. Defaults to `FALSE`

`twoSidedTest` logical. Return a two sided p-value for the ACE. Defaults to `TRUE`

`verbose` logical; prints dots when bootstrapping to show that something is happening. Bootstrapping can take a long time.

`isSlaveMode` logical. Internal Use only. Used in recursion.

Details

Performs a sensitivity analysis estimating the average causal effect among those who would have been selected regardless of treatment assignment (ACE) without assuming monotonicity (i.e., that one of the principal strata is empty). The method assumes no interference (i.e., potential outcomes of all subjects are unaffected by treatment assignment of other subjects) and ignorable (i.e., random) treatment assignment. ACE is identified by assuming values for the sensitivity parameters `beta0`, `beta1`, and one of the parameters `phi`, `psi`, or `Pi`. The sensitivity parameters `beta0` and `beta1` have a log-odds ratio interpretation (see help for `sensitivityGBH`).

Only one of the parameters `phi`, `psi`, or `Pi` should be specified as all depend on each other. `psi` is unrestrained taking any value on the real line. The other parameters, `psi` and `Pi` have constraints and there will be estimation problems if these parameters are set at values outside the of their range of acceptable values based on the observed data. See Shepherd, Gilbert, Dupont (in press) for more details.

Value

object of class `sensitivity3d`

`ACE` array; estimated values of ACE for all combinations of `beta0`, `beta1`, and `phi`, `Pi`, `psi`. Array dimensions are `length(beta0)`, `length(beta1)`, `length(psi)`.

`ACE.ci` array; confidence interval determined by `quantile` if `ci.method` includes "bootstrap". Otherwise calculated using analytic variance with large sample normal approximation. Array dimensions the same as `ACE` element.

`ACE.var` array; estimated variance of ACE. Array dimensions the same as `ACE` element.

`ACE.p` vector; estimated p-value of ACE.

`beta0` vector; β values used for the first group.

`alphahat0` vector; estimated α values for the first group.

`Fas0` function; estimator for the distribution function of *y0* in the first group in the always selected stratum.

`beta1` vector; β values used for the second group.

`alphahat1` vector; estimated α values for the second group.

`Fas1` function; estimator for the distribution function of *y1* in the second group in the always selected stratum.

`phi` vector; phi values used.

`Pi` vector; Pi values used.

`psi` vector; psi values used.

`ci.map` list; mapping of confidence interval to quantile probability. Use numbers contained within as indices to the `SCE.ci` element.

Author(s)

Bryan E. Shepherd
Department of Biostatistics
Vanderbilt University

Charles Dupont
Department of Biostatistics
Vanderbilt University

References

Jemiai Y (2005), “Semiparametric Methods for Inferring Treatment Effects on Outcomes Defined Only if a Post-Randomization Event Occurs,” unpublished doctoral dissertation under the supervision of A. Rotnitzky, Harvard School of Public Health, Dept. of Biostatistics.

Shepherd BE, Redman MW, Ankerst DP (2008), “Does Finasteride affect the severity of prostate cancer? A causal sensitivity analysis,” Journal of the American Statistical Association 2008, 484, 1392-1404.

Shepherd BE, Gilbert PB, and Dupont CT, “Sensitivity analyses comparing time-to-event outcomes only existing in a subset selected postrandomization and relaxing monotonicity,” Biometrics, in press.

See Also

sensitivityGBH, sensitivitySGD

Examples

```
data(vaccine.trial)
ansJR<-with(vaccine.trial,
          sensitivityJR(z=treatment,s=hiv.outcome,y=logVL,
                    beta0=c(-1,-.5,0,.5,1),
                    beta1=c(-1,-.5,0,.5,1),
                    phi=c(0.95,0.9), selection="infected",
                    groupings=c("placebo","vaccine"),
                    N.boot=100)
         )
ansJR

data(vaccine.trial)
ansJR<-with(vaccine.trial,
          sensitivityJR(z=treatment,s=hiv.outcome,y=logVL,
                    beta0=c(-1,-.5,0,.5,1),
                    beta1=c(-1,-.5,0,.5,1),
                    phi=c(0.95,0.9), selection="infected",
                    groupings=c("placebo","vaccine"),
                    custom.FUN=function(mu0, mu1, ...) mu1 - mu0,
                      upperTest=TRUE, lowerTest=TRUE, twoSidedTest=TRUE,
                      N.boot=100)
         )
ansJR
```

`sensitivitySGD` *principal stratification sensitivity analysis with time to event data relaxing monotonicity assumption.*

Description

Principal stratification sensitivity analysis with time to event data relaxing monotonicity as described by Shepherd, Gilbert, and Dupont (in press).

Usage

```
sensitivitySGD(z, s, d, y, v, beta0, beta1, phi, Pi, psi, tau,
               time.points, selection, trigger, groupings,
               followup.time,
               ci=0.95, ci.method = c("bootstrap", "analytic"),
               ci.type="twoSided", custom.FUN = NULL, na.rm = FALSE,
               N.boot = 100L, N.events = NULL, interval = c(-100, 100),
               upperTest = FALSE, lowerTest = FALSE, twoSidedTest=TRUE,
               inCore = TRUE,verbose = getOption("verbose"),
               colsPerFile = 1000L, isSlaveMode = FALSE)
```

Arguments

`z` vector; contains the grouping values (e.g., treatment assignment) for each record.

`s` vector; indicates whether a record is selected.

`d` vector; indicates whether a post-selection event has occurred. Can be `NA` for unselected records.

`y` vector; the length of time from selection until event (d) or censoring. Can be `NA` for unselected records.

`v` numeric vector; the length of time from randomization until selection or censoring.

`beta0` numeric vector; values of the sensitivity parameter β linking outcome in group *g0* with `selection` if assigned group *g1*.

`beta1` numeric vector; values of the sensitivity parameter β linking outcome in group *g1* with `selection` if assigned group *g0*.

`phi, Pi, psi` vectors; sensitivity parameters specifying the joint distribution of $S(g0)$, $S(g1)$. Only one of the three parameters should be specified. `psi` is the log-odds ratio of selection. `Pi` is the probability of being in the always selected principal stratum ($Pr(S(g0) = S(g1) = selected)$). `phi` is the probability of selection in group *g0* given selection in group *g1* ($Pr(S(g0) = 1|S(g1) = 1)$).

`tau` maximum observed follow-up time after selection. Selection weights are constant for $t >$ `tau`.

`time.points` vector; time points, *t*, at which $SCE(t)$ will be estimated.

`selection` The value of `s` indicating selection.

`trigger` The value of `d` that denotes the post-selection event.

`groupings` Vector of two elements `c(`*g0*`,`*g1*`)`, the first element *g0* being the value of `z` the delineates the first group, the last element *g1* being the value of `z` which delineates the second group.

`followup.time` numeric value; cut-off point for `v` after which records are lost to censoring.

`ci` numeric vector; confidence interval level, defaults to `0.95`.

`ci.method` character; method by which the confidence interval and variance are calculated. Can be "analytic" or "bootstrap". Currently only works for "bootstrap".

`ci.type` character vector; type of confidence interval that the corresponding `ci` element is refering to. Can be "upper", "lower", or "twoSided". Defaults to `"twoSided"`.

`custom.FUN` function; function to calculate custom result. `Fas0`, `Fas1`, `time.points`, `p0`, `p1` are available to be used as arguments in the custom function. The custom function must return a vector of elements that is the same length as `time.points`.

`na.rm` logical; indicates whether records that are invalid due to `NA` values should be removed from the data set.

`N.boot` integer; number of bootstrap repetitions that will be run when `ci.method` includes "bootstrap".

`N.events` integer; number of selection-events (S) for each bootstrap replication when doing selection-event based bootstrapping.

`interval` numeric vector of length 2. Controls the range limits used to by `optimize` to estimate α.

`lowerTest` logical. Return the lower one sided p-value for SCE. Defaults to `FALSE`

`upperTest` logical. Return the upper one sided p-value for SCE. Defaults to `FALSE`

`twoSidedTest` logical. Return a two sided p-value for SCE. Defaults to `TRUE`

`verbose` logical; prints dots when bootstrapping to show that something is happening. Bootstrapping can take a long time.

`inCore` logical; running in memory if `TRUE`, running with scratch files if `FALSE`. Default is `TRUE`. For large data analysis, the user may want to switch this to `FALSE` to allow for processing on data sets larger than can fit in memory.

`colsPerFile` integer; number of columns of the scratch file to process in each pass (e.g., 100 columns).

`isSlaveMode` logical. Internal Use only. Used in recursion.

Details

Performs a sensitivity analysis estimating the "survival causal effect" among those who would have been selected regardless of treatment assignment (SCE) without assuming monotonicity (i.e., that one of the principal stratum is empty). The method assumes no interference (i.e., potential outcomes of all subjects are unaffected by treatment assignment of other subjects), ignorable (i.e., random) treatment assignment, and independent censoring (i.e., time from selection to event is independent of time from selection until censoring). SCE is identified by assuming values for the sensitivity parameters `beta0`, `beta1`, and one of the parameters `phi`, `psi`, or `Pi`. The sensitivity parameters `beta0` and `beta1` have a log-odds ratio interpretation (see help for `sensitivityGBH`). Given selection in one treatment arm, the probability of selection if in the other treatment arm is assumed to be constant for for $T(\texttt{z}) >$ `tau`.

Only one of the parameters `phi`, `psi`, or `Pi` should be specified as all depend on each other. `psi` is unrestrained taking any value on the real line. The other parameters, `phi` and `Pi` have constraints and there will be estimation problems if these parameters are set at values outside the of their range of acceptable values based on the observed data. See Shepherd, Gilbert, Dupont (in press) for more details.

Value

object of class `sensitivity3d`

`SCE` array; Calculated values of SCE for all combinations of the values from `beta0`, `beta1`, `phi/Pi/psi`, and `time.points`. Array dimensions are `length(time.points)`, `length(beta0)`, `length(beta1)`, `length(psi)`.

`SCE.ci` array; Confidence interval of the SCE value. Confidence interval determined by `quantile` if using `ci.method` “bootstrap”. Otherwise calculated using analytic variance with large sample normal approximation. Array dimensions the same as element `SCE`.

`SCE.var` array; estimated variance of SCE. Array dimensions the same as element `SCE`.

`beta0` vector; β values used for first group.

`beta1` vector; β values used for second group.

`psi` vector; ψ values used.

`Pi` vector; Pi values used.

`psi` vector; psi values used.

`ci.map` list; mapping of confidence interval to quantile probability. Use numbers contained within as indices to the `SCE.ci` element.

Author(s)

Bryan E. Shepherd
Department of Biostatistics
Vanderbilt University

Charles Dupont
Department of Biostatistics
Vanderbilt University

References

Shepherd BE, Gilbert PB, and Dupont CT, "Sensitivity analyses comparing time-to-event outcomes only existing in a subset selected postrandomization and relaxing monotonicity," Biometrics, in press.

See Also

sensitivitySGL, sensitivityJR, Surv

Examples

```
data(vaccine.trial)
sens.analysis<-with(vaccine.trial,
                sensitivitySGD(z=treatment, s=hiv.outcome, y=followup.yearsART,
                          d=ARTinitiation, beta0=c(0,-.25,-.5),
                          beta1=c(0, -.25, -.5), phi=c(0.95, 0.90), tau=3,
                          time.points=c(2,3), selection="infected",
                          trigger="initiated ART",
                          groupings=c("placebo","vaccine"), ci=.95,
                          ci.method="bootstrap", N.boot=100)
               )
sens.analysis


sens.analysis2<-with(vaccine.trial,
                sensitivitySGD(z=treatment, s=hiv.outcome, y=followup.yearsART,
                          d=ARTinitiation, beta0=c(0,-.25,-.5),
                          beta1=c(0, -.25, -.5), phi=c(0.95, 0.90), tau=3,
                          time.points=c(2,3), selection="infected",
                          trigger="initiated ART",
                          groupings=c("placebo","vaccine"), ci=.95,
                          custom.FUN=function(Fas0,Fas1,...,time.points) {
                            Fas0(time.points) - Fas1(time.points)
                          },
                          ci.method="bootstrap", N.boot=100)
               )
sens.analysis2
```

sensitivitySGL *principal stratification sensitivity analysis with time to event data*

Description

Principal stratification sensitivity analysis with time to event data using the method described by Shepherd, Gilbert, and Lumley (2007).

Usage

```
sensitivitySGL(z, s, d, y, v, beta, tau, time.points, selection, trigger,
               groupings, empty.principal.stratum, followup.time,
               ci=0.95, ci.method = c("analytic", "bootstrap"),
               ci.type="twoSided", custom.FUN = NULL, na.rm = FALSE,
               N.boot = 100L, interval = c(-100, 100),
               upperTest = FALSE, lowerTest = FALSE, twoSidedTest = TRUE,
               verbose = getOption("verbose"), isSlaveMode = FALSE)
```

Arguments

`z` vector; contains the grouping values (e.g., treatment assignment) for each record.

`s` vector; indicates whether a record is selected.

`d` vector; indicates whether a post-selection event has occurred. Can be `NA` for unselected records.

`y` vector; the length of time from selection until event (d) or censoring. Can be `NA` for unselected records.

`v` numeric vector; the length of time from randomization until selection or censoring.

`beta` vector; values of the sensitivity parameter β. `Inf` and `-Inf` are acceptable.

`tau` maximum observed follow-up time after selection. Selection weights are constant for $t >$ `tau`.

`time.points` vector; time points, *t*, at which $SCE(t)$ will be estimated.

`selection` The value of `s` indicating selection.

`trigger` logical; the value of `d` that denotes the post-selection event.

`groupings` Vector of two elements `c(`*g0*`,`*g1*`)`, the first element *g0* being the value of `z` the delineates the first group, the last element *g1* being the value of `z` which delineates the second group.

`empty.principal.stratum`
vector of two elements `c(`*s0*`,`*s1*`)`; describes the `s` values that select the empty principal stratum. If `empty.principal.stratum=c(`*s0*`,`*s1*`)`, then stratum defined by $S(g0) = s0$ and $S(g1) = s1$ is the empty stratum. In this example *s0* and *s1* refer to the two possible values of `s`. (Note: method only works if $s0 \neq s1$).

`followup.time` numeric value; cut-off point for `v` after which records are lost to censoring.

`ci` numeric vector; confidence interval level, defaults to `0.95`.

`ci.method` character; method by which the confidence interval and variance are calculated. Can be “analytic” or “bootstrap”.

`ci.type` character vector; type of confidence interval that the corresponding `ci` element is referring to. Can be “upper”, “lower”, or “twoSided”. Defaults to `"twoSided"`.

`custom.FUN` function; function to calculate custom result. `Fas0`, `Fas1`, `time.points`, `p0`, `p1` are available to be used as arguments in the custom function. The custom function must return a vector of elements that is the same length as `time.points`.

`na.rm` logical; indicates whether records that are invalid due to `NA` values should be removed from the data set.

`N.boot` integer; number of bootstrap repetitions that will be run when `ci.method` includes “bootstrap”.

`interval` numeric vector of length 2. Controls the range limits used to by optimize to estimate α.

`lowerTest` logical; Return the lower one sided p-value for SCE. Defaults to `FALSE`

`upperTest` logical; Return the upper one sided p-value for SCE. Defaults to `FALSE`

`twoSidedTest` logical; Return a two sided p-value for SCE. Defaults to `TRUE`

`verbose` logical; prints dots when bootstrapping to show that something is happening. Bootstrapping can take a long time.

`isSlaveMode` logical. Internal Use only. Used in recursion.

Details

Performs a sensitivity analysis estimating the “survival causal effect” among those who would have been selected regardless of treatment assignment (SCE). The method assumes no interference (i.e., potential outcomes of all subjects are unaffected by treatment assignment of other subjects), ignorable (i.e., random) treatment assignment, monotonicity (i.e., one of the principal strata is empty), and independent censoring (i.e., time from selection to event is independent of time from selection until censoring). SCE is then identified by assuming a value of the sensitivity parameter β, where e^{β} has an odds ratio interpretation (see help for `sensitivityGBH`). Given selection in one treatment arm, the probability of selection if in the other treatment arm is assumed to be constant for for $T(\mathtt{z}) >$ `tau`.

SCE is computed at user specified time points.

Specifying `beta=-Inf` or `beta=Inf` estimates the bounds for SCE.

Value

object of class `sensitivity2d`

`SCE` $SCE(t) = Pr(T(g0) \leq t|S(g0) = S(g1) = \mathtt{selection}) - Pr(T(g1) \leq t|S(g0) = S(g1) = \mathtt{selection})$. Array of the estimated SCE at all `time.points` for specified beta values. Array dimensions are `length(time.points)` by `length(beta)`.

`SCE.ci` array; confidence interval of SCE determined by `quantile` if using `ci.method` includes “bootstrap”. Otherwise calculated using analytic variance with large sample normal approximation. Array dimensions the same as element `SCE`.

`SCE.var` array; estimated variance of SCE. Array dimensions the same as element `SCE`.

`ci.map` list; mapping of confidence interval to quantile probability. Use numbers contained within as indices to the `SCE.ci` element.

`beta` vector of user-specified β values

`alphahat` vector of estimated values of α

`y0` vector of unique event times in the first group.

`Fas0` matrix of estimated empirical distribution function values for `y0` in the first group in the always selected principal stratum. $Pr(Y(g0) \leq y0|S(g0) = S(g1) = \texttt{selection}; \beta)$

`y1` vector of unique event times in the second group.

`Fas1` matrix of estimated empirical distribution function values for `y1` in the second group in the always selected principal stratum. $Pr(Y(g1) \leq y1|S(g0) = S(g1) = \texttt{selection}; \beta)$

Author(s)

Bryan E. Shepherd
Department of Biostatistics
Vanderbilt University

Charles Dupont
Department of Biostatistics
Vanderbilt University

References

Shepherd BE, Gilbert PB, Lumley T (2007), “Sensitivity analyses comparing time-to-event outcomes existing only in a subset selected postrandomization,” Journal of the American Statistical Association 102, 573-582.

See Also

`sensitivityGBH`, `sensitivityHHS`, `sensitivitySGD`, `Surv`

Examples

```
data(vaccine.trial)
sens.time<-with(vaccine.trial,
                sensitivitySGL(z=treatment, s=hiv.outcome, y=followup.yearsART,
                          d=ARTinitiation, beta=c(.25, 0,-.25), tau=3,
                          time.points=c(2,3), selection="infected",
                          trigger="initiated ART", groupings=c("placebo","vaccine"),
                          empty.principal.stratum=c("not infected","infected"),
                          N.boot=50, interval=c(-200,200),
                          upperTest=TRUE, lowerTest=TRUE, twoSidedTest=TRUE)
               )
sens.time

sens.time2<-with(vaccine.trial,
                sensitivitySGL(z=treatment, s=hiv.outcome, y=followup.yearsART,
                          d=ARTinitiation, beta=c(.25, 0,-.25), tau=3,
                          time.points=c(2,3), selection="infected",
                          trigger="initiated ART", groupings=c("placebo","vaccine"),
                          empty.principal.stratum=c("not infected","infected"),
                    custom.FUN=function(Fas0,Fas1,time.points,
                ...) { Fas0(time.points) - Fas1(time.points) },
                    N.boot=50, interval=c(-200,200))
               )
sens.time2

sens.time3<-with(vaccine.trial,
                sensitivitySGL(z=treatment, s=hiv.outcome, y=followup.yearsART,
                    d=ARTinitiation, beta=c(-Inf,.25,0,-.25,Inf),
                    tau=3, time.points=c(2,3), selection="infected",
                    trigger="initiated ART", groupings=c("placebo","vaccine"),
                    empty.principal.stratum=c("not infected","infected"),
                    custom.FUN=function(Fas0,Fas1,time.points,
                ...) { Fas0(time.points) - Fas1(time.points) },
                    N.boot=50, interval=c(-200,200))
               )
sens.time3
```

vaccine.trial *Simulated Vaccine Trial Data*

Description

Simulated vaccine trial data for use in demonstrating the use of the sensitivity functions implemented in this package.

Usage

```
data(vaccine.trial)
```

Format

A data frame with 2000 observations on the following 5 variables.

`treatment` a factor with levels “placebo” ,“vaccine”

`hiv.outcome` a factor with levels “infected”, “not infected”

`logVL` a numeric vector

`ARTinitiation` a factor with levels “initiated ART”, “no ART”

`followup.yearsART` a numeric vector

Examples

```
set.seed(1063917538)
N<-2000
p0<-0.10
z<-c(rep(0,N/2),rep(1,N/2))
s0<-rbinom(N,1,p0)
y0<-rnorm(N,4.5,.75)
```

```
delta<-0
y1<-y0+delta
alpha<--4
beta<-1
w<-exp(alpha+beta*y0)/(1+exp(alpha+beta*y0))
s1<-s0*rbinom(N,1,w)
s<-s0*(1-z)+s1*z
y<-ifelse(s*(1-z)==1,y0,
    ifelse(s*z==1,y1,NA))
tjunk<-rexp(N,1/3)
cjunk<-runif(N,0,15)
t<-ifelse(s*(1-z)==1,tjunk,
    ifelse(s*z==1,tjunk,NA))
c1<-ifelse(s*(1-z)==1,cjunk,
     ifelse(s*z==1,cjunk,NA))
c<-pmin(c1,3)



treatment<-ifelse(z==1,"vaccine","placebo")
hiv.outcome<-ifelse(s==1,"infected","not infected")
logVL<-y
ARTinitiation<-ifelse(t<c,"initiated ART","no ART")
followup.yearsART<-round(pmin(t,c),2)


vaccine.trial<-data.frame(treatment=treatment,
                          hiv.outcome=hiv.outcome,
                          logVL=logVL, ARTinitiation=ARTinitiation,
                          followup.yearsART=followup.yearsART)
```

Index