Package ‘icensmis’

August 29, 2016

Type Package

Title Study Design and Data Analysis in the Presence of Error-Prone Diagnostic Tests and Self-Reported Outcomes

Version 1.3.1

Date 2015-10-10

Author Xiangdong Gu and Raji Balasubramanian

Maintainer Xiangdong Gu <ustcgxd@gmail.com>

Description We consider studies in which information from error-prone diagnostic tests or self-reports are gathered sequentially to determine the occurrence of a silent event. Using a likelihood-based approach incorporating the proportional hazards assumption, we provide functions to estimate the survival distribution and covariate effects. We also provide functions for power and sample size calculations for this setting.

License GPL (>= 2)

Imports Rcpp (>= 0.11.0)

LinkingTo Rcpp

Suggests testthat

NeedsCompilation yes

Repository CRAN

Date/Publication 2016-01-03 17:44:50

R topics documented:

datasim *Simulate data including multiple outcomes from error-prone diagnostic tests or self-reports*

Description

This function simulates a data of N subjects with misclassified outcomes, assuming each subject receives a sequence of pre-scheduled tests for disease status ascertainment. Each test is subject to error, characterized by sensitivity and specificity. An exponential distribution is assumed for the time to event of interest. Three kinds of covariate settings can be generated: one sample setting, two group setting, and continuous covariates setting with each covariate sampled from i.i.d. N(0, 1). Two missing mechanisms can be assumed, namely MCAR and NTFP. The MCAR setting assumes that each test is subject to a constant, independent probability of missingness. The NTFP mechanism includes two types of missingness - (1) incorporates a constant, independent, probability of missing for each test prior to the first positive test result; and (2) all test results after first positive are missing. The simulated data is in longitudinal form with one row per test time.

Covariate values, by default, are assumed to be constant. However, this function can simulate a special case of time varying covariates. Under time varying covariates setting, each subject is assumed to have a change time point, which is sampled from the visit times. We assume that each subject has two sets of covariate values. Before his change time point, the covariate values take from the first set, and second set after change time point. Thus, each subject's distribution of survival time is two-piece exponential distribution with different hazard rates.

Usage

```
datasim(N, blambda, testtimes, sensitivity, specificity, betas = NULL,
  twogroup = NULL, pmiss = 0, pcensor = 0, design = "MCAR",
  negpred = 1, time.varying = F)
```

Arguments

N	total number of subjects to be simulated
blambda	baseline hazard rate
testtimes	a vector of pre-scheduled test times
sensitivity	the sensitivity of test
specificity	the specificity of test
betas	a vector of regression coefficients of the same length as the covariate vector. If betas = NULL then the simulated dataset corresponds to the one sample setting. If betas != NULL and twogroup != NULL then the simulated dataset corresponds to the two group setting, and the first value of betas is used as the coefficient for the treatment group indicator. If betas != NULL and twogroup = NULL, then the covariates are ~ i.i.d. N(0, 1), and the number of covariates is determined by the length of betas.

`twogroup` corresponds to the proportion of subjects allocated to the baseline (reference) group in the two-group setting. For the two-group setting, this variable should be between 0 and 1. For the one sample and multiple (>= 2) covariate setting, this variable should be set to NULL. That is, when betas !=NULL, set twogroup to equal the proportion of the subjects in the baseline group to obtain a simulated dataset corresponding to the two-group setting. Else, set twogroup=NULL to obtain either the one sample setting (betas=NULL) or continuous covariates (betas !=NULL).

`pmiss` a value or a vector (must have same length as testtimes) of the probabilities of each test being randomly missing at each test time. If pmiss is a single value, then each test is assumed to have an identical probability of missingness.

`pcensor` a value or a vector (must have same length as testtimes) of the probability of censoring at each visit, assuming censoring process is independent on other missing mechanisms.

`design` missing mechanism: "MCAR" or "NTFP"

`negpred` baseline negative predictive value, i.e. the probability of being truely disease free for those who were tested (reported) as disease free at baseline. If baseline screening test is perfect, then negpred = 1.

`time.varying` indicator whether fitting a time varying covariate model or not

Details

To simulate the one sample setting data, set betas to be NULL. To simulate the two group setting data, set twogroup to equal the proportion of the subjects in the baseline group and set betas to equal the coefficient corresponding to the treatment group indicator(i.e. beta equals the log hazard ratio of the two groups). To simulate data with continuous i.i.d. N(0, 1) covariates, set twogroup to be NULL and set betas to equal the vector of coefficients of the covariates.

Value

simulated longitudinal form data frame

Examples

```
## One sample setting
simdata1 <- datasim(N = 1000, blambda = 0.05, testtimes = 1:8, sensitivity = 0.7,
  specificity = 0.98, betas = NULL, twogroup = NULL, pmiss = 0.3, design = "MCAR")

## Two group setting, and the two groups have same sample sizes
simdata2 <- datasim(N = 1000, blambda = 0.05, testtimes = 1:8, sensitivity = 0.7,
  specificity = 0.98, betas = 0.7, twogroup = 0.5, pmiss = 0.3, design = "MCAR")

## Three covariates with coefficients 0.5, 0.8, and 1.0
simdata3 <- datasim(N = 1000, blambda = 0.05, testtimes = 1:8, sensitivity = 0.7,
  specificity = 0.98, betas = c(0.5, 0.8, 1.0), twogroup = NULL, pmiss = 0.3,
  design = "MCAR", negpred = 1)

## NTFP missing mechanism
simdata4 <- datasim(N = 1000, blambda = 0.05, testtimes = 1:8, sensitivity = 0.7,
  specificity = 0.98, betas = c(0.5, 0.8, 1.0), twogroup = NULL, pmiss = 0.3,
  design = "NTFP", negpred = 1)

## Baseline misclassification
simdata5 <- datasim(N = 2000, blambda = 0.05, testtimes = 1:8, sensitivity = 0.7,
  specificity = 0.98, betas = c(0.5, 0.8, 1.0), twogroup = NULL, pmiss = 0.3,
  design = "MCAR", negpred = 0.97)

## Time varying covariates
simdata6 <- datasim(N = 1000, blambda = 0.05, testtimes = 1:8, sensitivity = 0.7,
  specificity = 0.98, betas = c(0.5, 0.8, 1.0), twogroup = NULL, pmiss = 0.3,
  design = "MCAR", negpred = 1, time.varying = TRUE)
```

`icmis` *Maximum likelihood estimation for settings of error-prone diagnostic tests and self-reported outcomes*

Description

This function estimates the baseline survival function evaluated at each test time in the presence of error-prone diagnostic tests and self-reported outcomes. If there are covariates included in the dataset, it also estimates their coefficients assuming proportional hazards. The covariate values can be either time independent or time varying The function can also be used to incorporate misclassification of disease status at baseline (due to an error-prone diagnostic procedure).

Usage

```
icmis(subject, testtime, result, data, sensitivity, specificity,
  formula = NULL, negpred = 1, time.varying = F, betai = NULL,
  initsurv = 0.5, param = 1, ...)
```

Arguments

`subject` variable in data for subject id.

`testtime` variable in data for test time. Assume all test times are non-negative. testtime = 0 refers to baseline visit (only used/needed if the model is time varying covarites)

`result` variable in data for test result.

`data` the data to analyze.

`sensitivity` the sensitivity of test.

`specificity` the specificity of test.

`formula` a formula to specify what covariates to be included in the model. If there is no covariate or one sample setting, set it to NULL. Otherwise, input like ~x1 + x2 + factor(x3).

`negpred` baseline negative predictive value, i.e. the probability of being truely disease free for those who were tested (reported) as disease free at baseline. If baseline screening test is perfect, then negpred = 1.

`time.varying` indicator whether fitting a time varying covariate model or not.

`betai` a vector of initial values for the regression coefficients corresponding to the vector of covariates. If betai=NULL, then 0s are used for the initial values. Otherwise, the length of betai must equal the number of covariates.

`initsurv` initial value for survival function of baseline group in the last visit time. It is used to compute initival values for survival function at all visit times.

`param` parameterization for survival function used for optimization, taking values 1, 2, or 3. There are 3 parameterizations available. param = 1: this parameterization uses the change in cumulative incidence in time period j for baseline group as parameters, i.e. log(S[j]) - log(S[j+1]). param = 2: simply use log of the parameters in param = 1 so that those parameters are unbounded. param = 3: the first element is log(-log(S[tau_1])) corresponding to log-log transformation of survival function at first visit, while other parameters are corresponding to the change in log-log of surival function, log(-log(S[j])) - log(-log(S[j-1])). In most cases, all parameters yield same results , while in some situations especially when two visit times are estimated to have same survival functions, they may differ. Choose the one that works best (check likelihood function)

`...` other arguments passed to `optim` function. For example, if the optimization does not converge, we can increase maxit in the optim function.

Details

The input data should be in longitudinal form with one row per test time. Use `datasim` to simulate a dataset to see the sample data structure. If time varying model is to be fitted, the baseline visit must be provided so that the baseline covariate information can be extracted. If an error is generated due to the optimization procedure, then we recommend trying different initial values.

This likelihood-based approach is a function of the survival function evaluated at each unique test time in the dataset and the vector of regression coefficients as model parameters. Therefore, it works best for situations where there is a limited number of unique test times in the dataset. If there are a large number of unique test times, one solution is to group several test times together.

Value

A list of fitting results is returned with log-likelihood, estimated coefficiets, estimated survival function, and estimated covariance matrix for covariates.

Examples

```
## One sample setting
simdata1 <- datasim(N = 1000, blambda = 0.05, testtimes = 1:8, sensitivity = 0.7,
  specificity = 0.98, betas = NULL, twogroup = NULL, pmiss = 0.3, design = "MCAR")
fit1 <- icmis(subject = ID, testtime = testtime, result = result, data = simdata1,
  sensitivity = 0.7, specificity= 0.98, formula = NULL, negpred = 1)

## Two group setting, and the two groups have same sample sizes
simdata2 <- datasim(N = 1000, blambda = 0.05, testtimes = 1:8, sensitivity = 0.7,
  specificity = 0.98, betas = 0.7, twogroup = 0.5, pmiss = 0.3, design = "MCAR")
fit2 <- icmis(subject = ID, testtime = testtime, result = result, data = simdata2,
  sensitivity = 0.7, specificity= 0.98, formula = ~group)

## Three covariates with coefficients 0.5, 0.8, and 1.0
simdata3 <- datasim(N = 1000, blambda = 0.05, testtimes = 1:8, sensitivity = 0.7,
  specificity = 0.98, betas = c(0.5, 0.8, 1.0), twogroup = NULL, pmiss = 0.3,
  design = "MCAR", negpred = 1)
fit3 <- icmis(subject = ID, testtime = testtime, result = result, data = simdata3,
  sensitivity = 0.7, specificity= 0.98, formula = ~cov1+cov2+cov3, negpred = 1)

## Fit data with NTFP missing mechanism (the fitting is same as MCAR data)
simdata4 <- datasim(N = 1000, blambda = 0.05, testtimes = 1:8, sensitivity = 0.7,
  specificity = 0.98, betas = c(0.5, 0.8, 1.0), twogroup = NULL, pmiss = 0.3,
  design = "NTFP", negpred = 1)
fit4 <- icmis(subject = ID, testtime = testtime, result = result, data = simdata4,
  sensitivity = 0.7, specificity= 0.98, formula = ~cov1+cov2+cov3, negpred = 1)

## Fit data with baseline misclassification
simdata5 <- datasim(N = 2000, blambda = 0.05, testtimes = 1:8, sensitivity = 0.7,
  specificity = 0.98, betas = c(0.5, 0.8, 1.0), twogroup = NULL, pmiss = 0.3,
  design = "MCAR", negpred = 0.97)
fit5 <- icmis(subject = ID, testtime = testtime, result = result, data = simdata5,
  sensitivity = 0.7, specificity= 0.98, formula = ~cov1+cov2+cov3, negpred = 0.97)

## Fit data with time varying covariates
simdata6 <- datasim(N = 1000, blambda = 0.05, testtimes = 1:8, sensitivity = 0.7,
  specificity = 0.98, betas = c(0.5, 0.8, 1.0), twogroup = NULL, pmiss = 0.3,
  design = "MCAR", negpred = 1, time.varying = TRUE)
fit6 <- icmis(subject = ID, testtime = testtime, result = result, data = simdata6,
   sensitivity = 0.7, specificity= 0.98, formula = ~cov1+cov2+cov3, negpred = 1,
   time.varying = TRUE)
```

---

icpower *Study design in the presence of error-prone diagnostic tests and self-reported outcomes*

Description

This function calculates the power and sample in the presence of error-prone diagnostic tests and self-reported outcomes. Two missing mechanisms can be assumed, namely MCAR and NTFP. The MCAR setting assumes that each test is subject to a constant, independent probability of missingness. The NTFP mechanism includes two types of missingness - (1) incorporates a constant, independent, probability of missing for each test prior to the first positive test result; and (2) all test results after first positive are missing.

Usage

```
icpower(HR, sensitivity, specificity, survivals, N = NULL, power = NULL,
  rho = 0.5, alpha = 0.05, pmiss = 0, pcensor = 0, design = "MCAR",
  negpred = 1)
```

Arguments

`HR` hazard ratio under the alternative hypothesis.

`sensitivity` the sensitivity of test.

`specificity` the specificity of test

`survivals` a vector of survival function at each test time for baseline(reference) group. Its length determines the number of tests.

`N` a vector of sample sizes to calculate corresponding powers. If one needs to calculate sample size, then set to NULL.

`power` a vector of powers to calculate corresponding sample sizes. If one needs to calculate power, then set to NULL.

`rho` proportion of subjects in baseline(reference) group.

`alpha` type I error.

`pmiss` a value or a vector (must have same length as survivals) of the probabilities of each test being randomly missing at each test time. If pmiss is a single value, then each test is assumed to have an identical probability of missingness.

`pcensor` a value or a vector (must have same length as testtimes) of the probability of censoring at each visit, assuming censoring process is independent on other missing mechanisms.

`design` missing mechanism: "MCAR" or "NTFP".

`negpred` baseline negative predictive value, i.e. the probability of being truely disease free for those who were tested (reported) as disease free at baseline. If baseline screening test is perfect, then negpred = 1.

Details

To calculate sample sizes for a vector of powers, set N = NULL. To calculate powers for a vector of sample sizes, set power = NULL. One and only one of power and N should be specified, and the other set to NULL. This function uses an enumeration algorithm to calculate the expected Fisher information matrix. The expected Fisher information matrix is used to obtain the variance of the coefficient corresponding to the treatment group indicator.

Value

- result: a data frame with calculated sample size and power
- I1 and I2: calculated unit Fisher information matrices for each group, which can be used to calculate more values of sample size and power for the same design without the need to enumerate again

Note

When diagnostic test is perfect, i.e. sensitivity=1 and specificity=1, use `icpowerpf` instead to obtain significantly improved computational efficiency.

See Also

`icpowerpf`

Examples

```
## First specificy survivals. Assume test times are 1:8, with survival function
## at the end time 0.9
surv <- exp(log(0.9)*(1:8)/8)

## Obtain power vs. N
pow1 <- icpower(HR = 2, sensitivity = 0.55, specificity = 0.99, survivals = surv,
   N = seq(500, 1500, 50), power = NULL, rho = 0.5, alpha = 0.05,
   pmiss = 0, design = "MCAR", negpred = 1)

plot(pow1$result$N, pow1$result$power, type="l", xlab="N", ylab="power")

## Calculate sample size, assuming desired power is 0.9
pow2 <- icpower(HR = 2, sensitivity = 0.55, specificity = 0.99, survivals = surv,
   N = NULL, power = 0.9, rho = 0.5, alpha = 0.05, pmiss = 0, design = "MCAR",
   negpred = 1)

## When missing test is present with MCAR
pow3 <- icpower(HR = 2, sensitivity = 0.55, specificity = 0.99, survivals = surv,
   N = NULL, power = 0.9, rho = 0.5, alpha = 0.05, pmiss = 0.4, design = "MCAR",
   negpred = 1)

## When missing test is present with NTFP
pow4 <- icpower(HR = 2, sensitivity = 0.55, specificity = 0.99, survivals = surv,
   N = NULL, power = 0.9, rho = 0.5, alpha = 0.05, pmiss = 0.4, design = "NTFP",
   negpred = 1)

## When baseline misclassification is present
pow5 <- icpower(HR = 2, sensitivity = 0.55, specificity = 0.99, survivals = surv,
   N = NULL, power = 0.9, rho = 0.5, alpha = 0.05, pmiss = 0, design = "MCAR",
   negpred = 0.98)

## When test is  perfect and no missing test
pow6 <- icpower(HR = 2, sensitivity = 1, specificity = 1, survivals = surv,
   N = NULL, power = 0.9, rho = 0.5, alpha = 0.05, pmiss = 0, design = "MCAR",
   negpred = 1)

## Different missing probabilities at each test time
pow7 <- icpower(HR = 2, sensitivity = 0.55, specificity = 0.99, survivals = surv,
   N = NULL, power = 0.9, rho = 0.5, alpha = 0.05, pmiss = seq(0.1, 0.8, 0.1),
   design = "MCAR")
```

`icpower.val` *Study design in the presence of error-prone diagnostic tests and self-reported outcomes when sensitivity and specificity are unkonwn and a validation set is used*

Description

This function calculates the power and sample size in the presence of error-prone diagnostic tests and self-reported outcomes when both sensitivity and specificity are unknown. In this case, a subject of the subjects receive both gold standard test and error-prone test at each non-missing visit. The remaining subjects receive only error-prone test. Here, for the validation set, NTFP refers to no test after first positive result from the gold standard test. Both sensitivity and specificity are treated as unknown parameters in this setting.

Usage

```
icpower.val(HR, sensitivity, specificity, survivals, N = NULL, power = NULL,
  rhoval, rho = 0.5, alpha = 0.05, pmiss = 0, design = ”MCAR”,
  designval = ”MCAR”, negpred = 1)
```

Arguments

`HR` hazard ratio under the alternative hypothesis.

`sensitivity` the sensitivity of test.

`specificity` the specificity of test

`survivals` a vector of survival function at each test time for baseline(reference) group. Its length determines the number of tests.

`N` a vector of sample sizes to calculate corresponding powers. If one needs to calculate sample size, then set to NULL.

`power` a vector of powers to calculate corresponding sample sizes. If one needs to calculate power, then set to NULL.

`rhoval` proportion of subjects in validation set.

`rho` proportion of subjects in baseline(reference) group.

`alpha` type I error.

`pmiss` a value or a vector (must have same length as survivals) of the probabilities of each test being randomly missing at each test time. If pmiss is a single value, then each test is assumed to have an identical probability of missingness.

`design` missing mechanism: "MCAR" or "NTFP".

`designval` missing mechanism of validation set: "MCAR" or "NTFP".

`negpred` baseline negative predictive value, i.e. the probability of being truely disease free for those who were tested (reported) as disease free at baseline. If baseline screening test is perfect, then negpred = 1.

Value

- result: a data frame with calculated sample size and power
- IR1 and IR2: calculated unit Fisher information matrices for each group in non-validation set
- IV1 and IV2: calculated unit Fisher information matrices for each group in validation set

Examples

```
surv <- exp(log(0.9)*(1:8)/8)
pow <- icpower.val(HR = 2, sensitivity = 0.55, specificity = 0.99,
   survivals = surv, power = 0.9, rhoval=0.05, design= ”NTFP”, designval = ”NTFP”)
pow$result
```

icpowerpf *Study design in the presence of interval censored outcomes (assuming perfect diagnostic tests)*

Description

This function implements power and sample size calculations for interval censored time-to-event outcomes, when the diagnostic tests are assumed to be perfect (i.e. sensitivity=1 and specificity=1). This is a special case of the more general study design function `icpower`. However, for the special case of perfect diagnostic tests, this function can be used with significantly improved computational efficiency.

Usage

```
icpowerpf(HR, survivals, N = NULL, power = NULL, rho = 0.5,
  alpha = 0.05, pmiss = 0)
```

Arguments

`HR`	hazard ratio under the alternative hypothesis.
`survivals`	a vector of survival function at each test time for baseline(reference) group. Its length determines the number of tests.
`N`	a vector of sample sizes to calculate corresponding powers. If one needs to calculate sample size, then set to NULL.
`power`	a vector of powers to calculate corresponding sample sizes. If one needs to calculate power, then set to NULL.
`rho`	proportion of subjects in baseline(reference) group.
`alpha`	type I error.
`pmiss`	a value or a vector (must have same length as survivals) of the probabilities of each test being randomly missing at each test time. If pmiss is a single value, then each test is assumed to have an identical probability of missingness.

Value

same form as returned value of `icpower`

Note

See `icpower` for more details in a general situation.

Examples

```
powpf1 <- icpowerpf(HR =2 , survivals = seq(0.9, 0.1, by=-0.1), N = NULL,
   power = 0.9, pmiss = 0)

powpf2 <- icpowerpf(HR =2 , survivals = seq(0.9, 0.1, by=-0.1), N = NULL,
   power = 0.9, pmiss = 0.7)

## Different missing probabilities at each test time
powpf3 <- icpowerpf(HR =2 , survivals = seq(0.9, 0.1, -0.1), N = NULL,
   power = 0.9, pmiss = seq(0.1, .9, 0.1))
```

Index