Package 'SurvImpute'

July 21, 2025

Type Package

Title Multiple Imputation for Missing Covariates in Time-to-Event Data

Version 0.1.0

Maintainer Qinghua Lian <qlian@mcw.edu>

Description
Generates multiple imputed datasets from a substantive model compatible fully conditional specification model for time-to-event data. Our method assumes that the censoring process also depends on the covariates with missing values. Details will be available in an upcoming publication.

License GPL-3

Imports survival, VGAM, MASS

Suggests mitools

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation no

Author Qinghua Lian [aut, cre, cph],
Soyoung Kim [aut],
Michael J. Martens [aut],
Kwang Woo Ahn [aut]

Repository CRAN

Date/Publication 2025-01-31 15:40:05 UTC

Contents

`generate.compt` *Simulate competing risks endpoints data where censoring depends on missing covariates.*

Description

We are assuming three covariates, two continuous and one binary, where we assume the first continous and the binary covariates have missing values, and the second continuous covariates is fully observed.

Usage

```
generate.compt(n, beta1, beta2, phi, gamma, seed)
```

Arguments

`n`	Sample Size.
`beta1`	Regression Coefficients for the event of interest process (length of 3)
`beta2`	Regression Coefficients for the event of competing risk process (length of 3).
`phi`	Regression Coefficients for the censoring process (length of 3).
`gamma`	Regression Coefficients for the missingness (length of 3).
`seed`	A random seed for data generation.

Value

A data frame with missing values.

Examples

```
# Simulate a data set with approximately 45% censoring, 35% cause 1,
# and 20% causes 2, and half of the subjects with at least one missing covariates.
generate.compt(n = 100, beta1 = c(1,1,-1), beta2 = c(2, 2, -2),
phi= c(-1,-1,-0.5), gamma = c(1,1,1,-1), seed = 112358)
```

`generate.surv` *Simulate survival endpoints data where censoring depends on missing covariates.*

Description

We are assuming three covariates, two continuous and one binary, where we assume the first continous and the binary covariates have missing values, and the second continuous covariates is fully observed.

Usage

```
generate.surv(n, beta, phi, gamma, seed)
```

Arguments

n	Sample Size.
beta	Regression Coefficients for the event process (length of 3).
phi	Regression Coefficients for the censoring process (length of 3).
gamma	Regression Coefficients for the missingness (length of 3).
seed	A random seed for data generation.

Value

A data frame with missing values. For the `delta` column, 0 = censored, and 1 = event.

Examples

```
# Simulate a data set with approximately 50% censoring
# and 50% of the subjects with at least one missing covariates.
generate.surv(n = 100, beta = c(1,1,-1), phi= c(1,-1,-0.5), gamma = c(3,2,-1), seed = 112358)
```

---

new.smcfcs.compt *Multiple imputation method for missing covariates in competing risks endpoint.*

Description

Generates multiple imputed datasets from a substantive model compatible fully conditional specification model. This method incorporates the cause-specific hazards regression model into the imputation stage, and assumes that the censoring process also depends on the covariates with missing values. Without loss of generality, we assumed there is only one competing event. Our method is an extension of Bartlett et. al. (2015) and Bartlett and Taylor (2016).

Usage

```
new.smcfcs.compt(data, smformula, method, m = 10, rjlimit = 5000)
```

Arguments

data	The input data with missing values. Note: all missing values should be coded as `"NA"`, all binary covariates should be coded as 0/1, and all categorical covariates with more than two categories should have a class of `"factor"`.
smformula	The substantive model formula. For competing risks data, this should be a list of two Cox models for both the event of interest and the competing event (see example).

method	A vector of strings for each covariate specifying the the type of regression model to impute them. The length of this vector should match the number of columns in the input dataset, and also match the position of each column (Including the outcomes). If a covariate is fully observed, the value should be blank (""). Other possible options are "norm"(linear regression for continuous covariates), "lm"(logistic regression for binary covariates), and "mlogit"(multinomial logistic regression for categorical covariates).
m	Number of complete datasets to generate, default is 10.
rjlimit	Maximum number of rejection sampling attempts, default is 5000. If there are subjects who did not get a success sampled value for the missing after reaching the limit, a warning message will be issued suggesting to increase the limit.

Value

A list containing the imputed datasets.

Author(s)

Qinghua Lian <qlian@mcw.edu>

References

Bartlett JW, Taylor JM. (2016) Missing covariates in competing risks analysis. *Biostatistics*. **17(4)**, 751-63.

Examples

```
# Generate data with missing values
compt <- generate.compt(n = 500, beta1 = c(1,1,-1), beta2 = c(2, 2, -2),
phi= c(-1,-1,-0.5), gamma = c(1,1,1,-1), seed = 112358)
# Impute
imputed <- new.smcfcs.compt(data = compt,
smformula = c("Surv(time, delta==1)~X1 +X2+X3", "Surv(time, delta==2)~X1 +X2+X3"),
method = c("","", "norm","logreg",""),m = 10, rjlimit = 10000)
# Fit a Cox regression on each imputed dataset, then produce the final estimates using Rubin's rule.
require(mitools)
library(survival)
imputed.fit <- with(imputationList(imputed), expr = coxph(Surv(time, delta==1) ~ X1+X2+X3))
summary(MIcombine(imputed.fit))
#' @author Qinghua Lian \email{qlian@mcw.edu}
```

---

new.smcfcs.surv *Multiple imputation method for missing covariates in survival endpoint.*

Description

Generates multiple imputed datasets from a substantive model compatible fully conditional specification model. This method incorporates the cause-specific hazards regression model into the imputation stage, and assumes that the censoring process also depends on the covariates with missing values. Without loss of generality, we assumed there is only one competing event. Our method is an extension of Bartlett et. al. (2015) and Bartlett and Taylor (2016).

Usage

```
new.smcfcs.surv(data, smformula, method, m = 10, rjlimit = 5000)
```

Arguments

`data`	The input data with missing values. Note: all missing values should be coded as `"NA"`, all binary covariates should be coded as 0/1, and all categorical covariates with more than two categories should have a class of `factor`.
`smformula`	The substantive model formula. For survival data, this should be a formula of `"Surv(time, delta) ~ Covariates list"` type. (see example).
`method`	A vector of strings for each covariate specifying the the type of regression model to impute them. The length of this vector should match the number of columns in the input dataset, and also match the position of each column (Including the outcomes). If a covariate is fully observed, the value should be blank (""). Other possible options are `"norm"`(linear regression for continuous covariates), `"lm"`(logistic regression for binary covariates), and `"mlogit"`(multinomial logistic regression for categorical covariates).
`m`	Number of complete datasets to generate, default is 10.
`rjlimit`	Maximum number of rejection sampling attempts, default is 5000. If there are subjects who did not get a success sampled value for the missing after reaching the limit, a warning message will be issued suggesting to increase the limit.

Value

A list containing the imputed datasets.

References

Bartlett JW, Seaman SR, White IR, Carpenter JR. (2015). Multiple imputation of covariates by fully conditional specification: accommodating the substantive model. *Statistical Methods in Medical Research*. **24(4)**, 462-487.

Examples

```
# Generate data with missing values
cox <- generate.surv(n = 500, beta = c(1,1,-1), phi= c(1,-1,-0.5), gamma = c(3,2,-1), seed = 112358)
# Impute
imputed <- new.smcfcs.surv(data = cox, smformula = "Surv(time, delta)~X1 +X2+X3",
method = c("","", "norm","logreg",""), m = 10, rjlimit = 10000)
# Fit a Cox regression on each imputed dataset, then produce the final estimates using Rubin's rule.
```

```
require(mitools)
library(survival)
imputed.fit <- with(imputationList(imputed), expr = coxph(Surv(time, delta) ~ X1+X2+X3))
summary(MIcombine(imputed.fit))
```

Index