Package 'smcfcs'

October 10, 2018

Title Multiple Imputation of Covariates by Substantive Model Compatible Fully Conditional Specification

Version 1.3.1

URL http://www.missingdata.org.uk, http://thestatsgeek.com

Description Implements multiple imputation of missing covariates by Substantive Model Compatible Fully Conditional Specification. This is a modification of the popular FCS/chained equations multiple imputation approach, and allows imputation of missing covariate values from models which are compatible with the user specified substantive model.

Depends R (>= 3.1.2)

License GPL-3

LazyData true

Imports MASS, survival, VGAM, stats

Suggests knitr, rmarkdown, mitools

VignetteBuilder knitr

RoxygenNote 6.1.0

NeedsCompilation no

Author Jonathan Bartlett [aut, cre], Ruth Keogh [aut]

Maintainer Jonathan Bartlett <j.w.bartlett@bath.ac.uk>

Repository CRAN

Date/Publication 2018-10-10 11:50:06 UTC

R topics documented:

ex_cc 2
ex_compet 3
ex_coxquad 3
ex_lininter 4

ex_linquad 4
ex_logisticquad 5
ex_ncc 5
ex_poisson 6
smcfcs 6
smcfcs.casecohort 10
smcfcs.nestedcc 12

Index **14**

`ex_cc` *Simulated case cohort data*

Description

A dataset containing simulated case cohort data, where the sub-cohort was a 10% random sample of the full cohort.

Usage

```
ex_cc
```

Format

A data frame with 1571 rows and 7 variables:

t Time to event or censoring

d Indicator of whether event 1 occurred (d=1), or not (d=0)

x Partially observed continuous covariate

z Fully observed covariate

in.subco A binary indicator of whether the subject is in the sub-cohort

id An id variable

entertime The entry time variable to be used in the analysis

ex_compet — *Simulated example data with competing risks outcome and partially observed covariates*

Description

A dataset containing simulated competing risks data. There are two competing risks, and some times are also censored.

Usage

```
ex_compet
```

Format

A data frame with 1000 rows and 4 variables:

t Time to event or censoring

d Indicator of whether event 1 occurred (d=1), event 2 occurred (d=2) or individual was censored (d=0)

x1 Partially observed binary covariate, with linear effects on log competing risk hazards

x2 Partially observed normally distributed (conditional on x1) covariate, with linear effects on log competing risk hazards

ex_coxquad — *Simulated example data with time to event outcome and quadratic covariate effects*

Description

A dataset containing simulated data where a time to event outcome depends quadratically on a partially observed covariate.

Usage

```
ex_coxquad
```

Format

A data frame with 1000 rows and 6 variables:

t Time to event or censoring

d Binary indicator of whether event occurred or individual was censored

z Fully observed covariate, with linear effect on outcome (on log hazard scale)

x Partially observed normally distributed covariate, with quadratic effect on outcome (on log hazard scale)

xsq The square of x, which thus has missing values also

v An auxiliary variable (i.e. not contained in the substantive model)

`ex_lininter` *Simulated example data with continuous outcome and interaction between two partially observed covariates*

Description

A dataset containing simulated data where the outcome depends on both main effects and interaction of two partially observed covariates.

Usage

```
ex_lininter
```

Format

A data frame with 1000 rows and 4 variables:

y Continuous outcome

x1 Partially observed normally distributed covariate

x2 Partially observed binary covariate

`ex_linquad` *Simulated example data with continuous outcome and quadratic covariate effects*

Description

A dataset containing simulated data where the outcome depends quadratically on a partially observed covariate.

Usage

```
ex_linquad
```

Format

A data frame with 1000 rows and 5 variables:

y Continuous outcome

z Fully observed covariate, with linear effect on outcome

x Partially observed normally distributed covariate, with quadratic effect on outcome

xsq The square of x, which thus has missing values also

v An auxiliary variable (i.e. not contained in the substantive model)

ex_logisticquad — *Simulated example data with binary outcome and quadratic covariate effects*

Description

A dataset containing simulated data where the binary outcome depends quadratically on a partially observed covariate.

Usage

```
ex_logisticquad
```

Format

A data frame with 1000 rows and 5 variables:

y Binary outcome

z Fully observed covariate, with linear effect on outcome (on log odds scale)

x Partially observed normally distributed covariate, with quadratic effect on outcome (on log odds scale)

xsq The square of x, which thus has missing values also

v An auxiliary variable (i.e. not contained in the substantive model)

ex_ncc — *Simulated nested case-control data*

Description

A dataset containing simulated nested case-control data.

Usage

```
ex_ncc
```

Format

A data frame with 728 rows and 8 variables:

t Time to event or censoring

d Indicator of whether event 1 occurred (d=1), or not (d=0)

x Partially observed binary covariate

z Fully observed covariate

id An id variable

numrisk Number of patients at risk at time of case's event

setno The case-control set number

case Binary indicator of case (=1) or control (=0)

`ex_poisson` *Simulated example data with count outcome, modelled using Poisson regression*

Description

A dataset containing simulated data where the count outcome depends on two covariates, x and z, with missing values in x. The substantive model is Poisson regression.

Usage

```
ex_poisson
```

Format

A data frame with 1000 rows and 3 variables:

y Count outcome

z Fully observed covariate, with linear effect on outcome

x Partially observed normally distributed covariate, with linear effect on outcome

---

`smcfcs` *Substantive model compatible fully conditional specification imputation of covariates.*

Description

Multiply imputes missing covariate values using substantive model compatible fully conditional specification.

Usage

```
smcfcs(originaldata, smtype, smformula, method, predictorMatrix = NULL,
  m = 5, numit = 10, rjlimit = 1000, noisy = FALSE)
```

Arguments

originaldata	The original data frame with missing values.
smtype	A string specifying the type of substantive model. Possible values are "lm", "logistic", "poisson", "coxph" and "compet".
smformula	The formula of the substantive model. For "coxph" substantive models the left hand side should be of the form "Surv(t,d)". For "compet" substantive models, a list should be passed consisting of the Cox models for each cause of failure (see example).
method	A required vector of strings specifying for each variable either that it does not need to be imputed (""), the type of regression model to be be used to impute. Possible values are "norm" (normal linear regression), "logreg" (logistic regression), "poisson" (Poisson regression), "podds" (proportional odds regression for ordered categorical variables), "mlogit" (multinomial logistic regression for unordered categorical variables), or a custom expression which defines a passively imputed variable, e.g. "x^2" or "x1*x2".
predictorMatrix	An optional predictor matrix. If specified, the matrix defines which covariates will be used as predictors in the imputation models (the outcome must not be included). The i'th row of the matrix should consist of 0s and 1s, with a 1 in the j'th column indicating the j'th variable be used as a covariate when imputing the i'th variable. If not specified, when imputing a given variable, the imputation model covariates are the other covariates of the substantive model which are partially observed (but which are not passively imputed) and any fully observed covariates (if present) in the substantive model. Note that the outcome variable is implicitly conditioned on by the rejection sampling scheme used by smcfcs, and should not be specified as a predictor in the predictor matrix.
m	The number of imputed datasets to generate. The default is 5.
numit	The number of iterations to run when generating each imputation. In a (limited) range of simulations good performance was obtained with the default of 10 iterations. However, particularly when the proportion of missingness is large, more iterations may be required for convergence to stationarity.
rjlimit	Specifies the maximum number of attempts which should be made when using rejection sampling to draw from imputation models. If the limit is reached when running a warning will be issued. In this case it is probably advisable to increase the rjlimit until the warning does not appear.
noisy	logical value (default FALSE) indicating whether output should be noisy, which can be useful for debugging or checking that models being used are as desired.

Details

smcfcs imputes missing values of covariates using the Substantive Model Compatible Fully Conditional Specification multiple imputation approach proposed by Bartlett *et al* 2015 (see references).

Currently imputation is supported for linear regression ("lm"), logistic regression ("logistic"), Poisson regression ("poisson"), Cox regression for time to event data ("coxph"), and Cox models for competing risks data ("compet"). For the latter, a Cox model is assumed for each cause

of failure, and the event indicator should be integer coded with 0 corresponding to censoring, 1 corresponding to failure from the first cause etc.

The function returns a list. The first element `impDataset` of the list is a list of the imputed datasets. Models (e.g. the substantive model) can be fitted to each and results combined using Rubin's rules using the mitools package, as illustrated in the examples.

The second element `smCoefIter` is a three dimensional array containing the values of the substantive model parameters obtained at the end of each iteration of the algorithm. The array is indexed by: imputation number, parameter number, iteration.

If the substantive model is linear, logistic or Poisson regression, `smcfcs` will automatically impute missing outcomes, if present, using the specified substantive model. However, even in this case, the user should specify "" in the element of method corresponding to the outcome variable.

The development of this package was supported by a UK Medical Research Council Fellowship (MR/K02180X/1). Part of its development took place while the author was kindly hosted by the University of Michigan's Department of Biostatistics & Institute for Social Research.

The structure of many of the arguments to `smcfcs` are based on those of the excellent `mice` package.

Value

A list containing:

`impDatasets` a list containing the imputed datasets

`smCoefIter` a three dimension matrix containing the substantive model parameter values. The matrix is indexed by [imputation,parameter number,iteration]

Author(s)

Jonathan Bartlett <j.w.bartlett@bath.ac.uk> http://www.missingdata.org.uk http://thestatsgeek.com

References

Bartlett JW, Seaman SR, White IR, Carpenter JR. Multiple imputation of covariates by fully conditional specification: accommodating the substantive model. Statistical Methods in Medical Research 2015; 24(4): 462-487. http://doi.org/10.1177/0962280214521348

Examples

```
#set random number seed to make results reproducible
set.seed(123)

#linear substantive model with quadratic covariate effect
imps <- smcfcs(ex_linquad, smtype="lm", smformula="y~z+x+xsq",
               method=c("","","norm","x^2",""))

#if mitools is installed, fit substantive model to imputed datasets
#and combine results using Rubin's rules
if (requireNamespace("mitools", quietly = TRUE)) {
  library(mitools)
  impobj <- imputationList(imps$impDatasets)
```

```
  models <- with(impobj, lm(y~z+x+xsq))
  summary(MIcombine(models))
}

#the following examples are not run when the package is compiled on CRAN
#(to keep computation time down), but they can be run by package users
## Not run:
  #examining convergence, using 100 iterations, setting m=1
  imps <- smcfcs(ex_linquad, smtype="lm", smformula="y~z+x+xsq",
                 method=c("","","norm","x^2",""),m=1,numit=100)
  #convergence plot from first imputation for third coefficient of substantive model
  plot(imps$smCoefIter[1,3,])

 #include auxiliary variable assuming it is conditionally independent of Y (which it is here)
  predMatrix <- array(0, dim=c(ncol(ex_linquad),ncol(ex_linquad)))
  predMatrix[3,] <- c(0,1,0,0,1)
  imps <- smcfcs(ex_linquad, smtype="lm", smformula="y~z+x+xsq",
                 method=c("","","norm","x^2",""),predictorMatrix=predMatrix)

  #impute missing x1 and x2, where they interact in substantive model
  imps <- smcfcs(ex_lininter, smtype="lm", smformula="y~x1+x2+x1*x2",
                 method=c("","norm","logreg"))

  #logistic regression substantive model, with quadratic covariate effects
  imps <- smcfcs(ex_logisticquad, smtype="logistic", smformula="y~z+x+xsq",
                 method=c("","","norm","x^2",""))

  #Poisson regression substantive model
  imps <- smcfcs(ex_poisson, smtype="poisson", smformula="y~x+z",
                 method=c("","norm",""))
  if (requireNamespace("mitools", quietly = TRUE)) {
    library(mitools)
    impobj <- imputationList(imps$impDatasets)
    models <- with(impobj, glm(y~x+z,family=poisson))
    summary(MIcombine(models))
  }

  #Cox regression substantive model, with only main covariate effects
  if (requireNamespace("survival", quietly = TRUE)) {
    imps <- smcfcs(ex_coxquad, smtype="coxph", smformula="Surv(t,d)~z+x+xsq",
                   method=c("","","","norm","x^2",""))

    #competing risks substantive model, with only main covariate effects
    imps <- smcfcs(ex_compet, smtype="compet",
                   smformula=c("Surv(t,d==1)~x1+x2", "Surv(t,d==2)~x1+x2"),
                   method=c("","","logreg","norm"))
  }

  #if mitools is installed, fit model for first competing risk
  if (requireNamespace("mitools", quietly = TRUE)) {
    library(mitools)
    impobj <- imputationList(imps$impDatasets)
    models <- with(impobj, coxph(Surv(t,d==1)~x1+x2))
```

```
  summary(MIcombine(models))
}


## End(Not run)
```

`smcfcs.casecohort` *Substantive model compatible fully conditional specification imputation of covariates for case cohort studies*

Description

Multiply imputes missing covariate values using substantive model compatible fully conditional specification for case cohort studies.

Usage

```
smcfcs.casecohort(originaldata, smformula, sampfrac, in.subco, method,
  predictorMatrix = NULL, m = 5, numit = 10, rjlimit = 1000,
  noisy = FALSE)
```

Arguments

`originaldata`	The case-cohort data set (NOT a full cohort data set with a case-cohort substudy within it)
`smformula`	A formula of the form "Surv(entertime,t,d)~x", where d is the event (d=1) or censoring (d=0) indicator, t is the event or censoring time and entertime is equal to the time origin (typically 0) for individuals in the subcohort and is equal to (t-0.001) for cases outside the subcohort [this sets cases outside the subcohort to enter follow-up just before their event time. The value 0.001 may need to be modified depending on the time scale.]
`sampfrac`	The proportion of individuals from the underlying full cohort who are in the subcohort
`in.subco`	The name of a column in the dataset with 0/1s that indicates whether the subject is in the subcohort
`method`	A required vector of strings specifying for each variable either that it does not need to be imputed (""), the type of regression model to be be used to impute. Possible values are `"norm"` (normal linear regression), `"logreg"` (logistic regression), `"poisson"` (Poisson regression), `"podds"` (proportional odds regression for ordered categorical variables), `"mlogit"` (multinomial logistic regression for unordered categorical variables), or a custom expression which defines a passively imputed variable, e.g. `"x^2"` or `"x1*x2"`.
`predictorMatrix`	An optional predictor matrix. If specified, the matrix defines which covariates will be used as predictors in the imputation models (the outcome must not be included). The i'th row of the matrix should consist of 0s and 1s, with a 1 in the

j'th column indicating the j'th variable be used as a covariate when imputing the i'th variable. If not specified, when imputing a given variable, the imputation model covariates are the other covariates of the substantive model which are partially observed (but which are not passively imputed) and any fully observed covariates (if present) in the substantive model. Note that the outcome variable is implicitly conditioned on by the rejection sampling scheme used by smcfcs, and should not be specified as a predictor in the predictor matrix.

`m` The number of imputed datasets to generate. The default is 5.

`numit` The number of iterations to run when generating each imputation. In a (limited) range of simulations good performance was obtained with the default of 10 iterations. However, particularly when the proportion of missingness is large, more iterations may be required for convergence to stationarity.

`rjlimit` Specifies the maximum number of attempts which should be made when using rejection sampling to draw from imputation models. If the limit is reached when running a warning will be issued. In this case it is probably advisable to increase the `rjlimit` until the warning does not appear.

`noisy` logical value (default FALSE) indicating whether output should be noisy, which can be useful for debugging or checking that models being used are as desired.

Details

This version of `smcfcs` is designed for use with case cohort studies but where the analyst does not wish to, or cannot (due to not having the necessary data) impute the full cohort. The function's arguments are the same as for the main smcfcs function, except for `smformula`, `in.subco`, and `sampfrac` - see above for details on how these should be specified.

Author(s)

Ruth Keogh <ruth.keogh@lshtm.ac.uk>

Jonathan Bartlett <j.w.bartlett@bath.ac.uk>

Examples

```
#the following example is not run when the package is compiled on CRAN
#(to keep computation time down), but it can be run by package users
## Not run:
  #as per the documentation for ex_cc, the sampling fraction is 10%
  imps <- smcfcs.casecohort(ex_cc, smformula="Surv(entertime, t, d)~x+z", sampfrac=0.1,
                            in.subco="in.subco", method=c("", "", "norm", "", "", "", ""))
  library(mitools)
  impobj <- imputationList(imps$impDatasets)
  models <- with(impobj, coxph(Surv(entertime,t,d)~x+z+cluster(id)))
  summary(MIcombine(models))

## End(Not run)
```

`smcfcs.nestedcc` *Substantive model compatible fully conditional specification imputation of covariates for nested case control studies*

Description

Multiply imputes missing covariate values using substantive model compatible fully conditional specification for nested case control studies.

Usage

```
smcfcs.nestedcc(originaldata, smformula, set, event, nrisk, method,
  predictorMatrix = NULL, m = 5, numit = 10, rjlimit = 1000,
  noisy = FALSE)
```

Arguments

`originaldata`	The nested case-control data set (NOT a full cohort data set with a case-cohort substudy within it)
`smformula`	A formula of the form "Surv(t,case)~x+strata(set)", where case is case-control indicator, t is the event or censoring time. Note that t could be set to the case's event time for the matched controls in a given set. The right hand side should include the case control set as a strata term (see example).
`set`	variable identifying matched sets in nested case-control study
`event`	variable which indicates who is a case/control in the nested case-control sample. Note that this is distinct from d.
`nrisk`	variable which is the number at risk (in the underlying full cohort) at the event time for the case in each matched set (i.e. nrisk is the same for all individuals in a matched set).
`method`	A required vector of strings specifying for each variable either that it does not need to be imputed (""), the type of regression model to be be used to impute. Possible values are `"norm"` (normal linear regression), `"logreg"` (logistic regression), `"poisson"` (Poisson regression), `"podds"` (proportional odds regression for ordered categorical variables), `"mlogit"` (multinomial logistic regression for unordered categorical variables), or a custom expression which defines a passively imputed variable, e.g. `"x^2"` or `"x1*x2"`.
`predictorMatrix`	An optional predictor matrix. If specified, the matrix defines which covariates will be used as predictors in the imputation models (the outcome must not be included). The i'th row of the matrix should consist of 0s and 1s, with a 1 in the j'th column indicating the j'th variable be used as a covariate when imputing the i'th variable. If not specified, when imputing a given variable, the imputation model covariates are the other covariates of the substantive model which are partially observed (but which are not passively imputed) and any fully observed covariates (if present) in the substantive model. Note that the outcome variable

	is implicitly conditioned on by the rejection sampling scheme used by smcfcs, and should not be specified as a predictor in the predictor matrix.
m	The number of imputed datasets to generate. The default is 5.
numit	The number of iterations to run when generating each imputation. In a (limited) range of simulations good performance was obtained with the default of 10 iterations. However, particularly when the proportion of missingness is large, more iterations may be required for convergence to stationarity.
rjlimit	Specifies the maximum number of attempts which should be made when using rejection sampling to draw from imputation models. If the limit is reached when running a warning will be issued. In this case it is probably advisable to increase the rjlimit until the warning does not appear.
noisy	logical value (default FALSE) indicating whether output should be noisy, which can be useful for debugging or checking that models being used are as desired.

Details

This version of smcfcs is designed for use with nested case control studies. The function's arguments are the same as for the main smcfcs function, except for smformula, set, event and nrisk - see above for details on how these should be specified.

Author(s)

Ruth Keogh <ruth.keogh@lshtm.ac.uk>

Jonathan Bartlett <j.w.bartlett@bath.ac.uk>

Examples

```
#the following example is not run when the package is compiled on CRAN
#(to keep computation time down), but it can be run by package users
## Not run:
  predictorMatrix <- matrix(0,nrow=dim(ex_ncc)[2],ncol=dim(ex_ncc)[2])
  predictorMatrix[which(colnames(ex_ncc)=="x"),c(which(colnames(ex_ncc)=="z"))] <- 1

  imps <- smcfcs.nestedcc(originaldata=ex_ncc,set="setno",nrisk="numrisk",event="d",
                          smformula="Surv(t,case)~x+z+strata(setno)",
                          method=c("", "", "logreg", "", "", "", "", ""),
                          predictorMatrix=predictorMatrix)
  library(mitools)
  impobj <- imputationList(imps$impDatasets)
  models <- with(impobj, clogit(case~x+z+strata(setno)))
  summary(MIcombine(models))

## End(Not run)
```

Index

∗Topic **datasets**
ex_cc, 2
ex_compet, 3
ex_coxquad, 3
ex_lininter, 4
ex_linquad, 4
ex_logisticquad, 5
ex_ncc, 5
ex_poisson, 6

ex_cc, 2
ex_compet, 3
ex_coxquad, 3
ex_lininter, 4
ex_linquad, 4
ex_logisticquad, 5
ex_ncc, 5
ex_poisson, 6

smcfcs, 6
smcfcs.casecohort, 10
smcfcs.nestedcc, 12