Package 'gerbil'

July 22, 2025

Type Package

Title Generalized Efficient Regression-Based Imputation with Latent Processes

Version 0.1.9

Author Michael Robbins [aut, cre],
Max Griswold [ctb],
Pedro Nascimento de Lima [ctb]

Maintainer Michael Robbins <mrobbins@rand.org>

Description Implements a new multiple imputation method that draws imputations from a latent joint multivariate normal model which underpins generally structured data. This model is constructed using a sequence of flexible conditional linear models that enables the resulting procedure to be efficiently implemented on high dimensional datasets in practice. See Robbins (2021) <doi:10.48550/arXiv.2008.02243>.

License GPL-2

Depends R (>= 2.10)

Imports base, DescTools, graphics, grDevices, lattice, MASS, mvtnorm, openxlsx, parallel, pbapply, stats, truncnorm, utils

Suggests dplyr, knitr, mice, rmarkdown, testthat (>= 2.1.0)

VignetteBuilder knitr

Encoding UTF-8

LazyData true

NeedsCompilation no

RoxygenNote 7.1.1

Repository CRAN

Date/Publication 2023-01-12 11:20:02 UTC

Contents

`cor_gerbil` *Correlation Analysis for* `gerbil` *Objects*

Description

This function assesses the bivariate properties of imputed data using a correlation analysis. Specifically, it calculates pairwise correlations for observed cases and for imputed cases. The function also calculates the Fisher z-transformation for each correlation and performs a hypothesis test using the transformed correlations in order to compare correlations calculated using imputed cases to those calculated using observed cases.

Usage

```
cor_gerbil(x, y = NULL, imp = 1, log = NULL, partial = "imputed")
```

Arguments

`x`	A `gerbil` object containing the imputed data.
`y`	A vector listing the column names of the imputed data that will be included in the correlation analysis. By default, `y` contains all columns of the data that required imputation. If `TRUE`, all variables with missing values eligible for imputation are used.
`imp`	A scalar indicating which of the multiply imputed datasets should be used for the analysis. Defaults to `imp = 1`.
`log`	A character vector that includes names variable of which a log transformation is to be taken prior to calculating correlations.
`partial`	Indicates how partially imputed pairs are handled when calculating correlations. If `partial = 'imputed'`, cases with at least one missing variable in a pair are considered imputed. Otherwise (`partial = 'observed'`), only cases with both variables in the pair missing are considered imputed.

Details

Cases are assigned a status of being observed or imputed in a pairwise fashion. That is, a specific data unit may be considered observed when calculating a correlation for one pair of variables and be imputed when calculating a correlation for another pair. For a given pair of variables, cases that have both variables observed are always treated as observed, and cases that have both variables missing are always treated as imputed. Cases that have only one variable in the pair observed (i.e., those that are partially imputed) are treated as imputed when the input `partial = 'imputed'` (the default) and are otherwise treated as observed.

Correlations are calculated across an expanded dataset that creates binary indicators for categorical variables and for semicontinuous variables. Unlike the algorithm used to calculate the imputations, missingness is not artificially imposed in any binary indicator. Missingness is imposed, however, in the variable corresponding to the continuous portion of a semicontinuous variable.

Note that the hypothesis test based upon the Fisher z-transformation is based off of bivariate normal assumptions. As such, p-values may be misleading in data where this assumption does not hold.

Value

`cor_gerbil()` retuns an object of the class `cor_gerbil` that has following slots:

Correlations A list containing two elements – these are named `Observed`, `Imputed`, and `All`. The first is a matrix giving the sample correlations when calculated across cases labeled as observed. The second and third are analogous correlation matrices calculated across only cases labeled as imputed and across all cases, respectively.

n A list containing two elements – these are named `Observed`, `Imputed`, and `All`. The first is a matrix giving number of cases in the respective pair of variables that have been labeled as observed. The second and third are analogous matrices indicating the number of cases labeled as imputed for each pair and indicating the total number of cases for each pair, respectively.

Fisher.Z A list containing two elements – these are named `Observed`, `Imputed`, and `All`. These matrices give the Fisher z-transformation of the correlations in the matrices provided in the slot `Correlations`.

Statistic A matrix that gives the value of the test statistic based on the Fisher z-transformation for each pair of variables. This statistic may be used to assess whether the correlations calculated across cases labeled as observed are statistically different from the correlations calculated across cases labeled as imputed.

p.value A matrix that list the p-value for each test statistic provided in the matrix in the slot labeled `Statistic`.

Examples

```
#Load the India Human Development Survey-II dataset
data(ihd_mcar)

imps.gerbil <- gerbil(ihd_mcar, m = 1, mcmciter = 100, ords = "education_level",
     semi = "farm_labour_days", bincat = c("sex", "marital_status", "job_field", "own_livestock"))

#Run the correlation analysis
cors.gerbil <- cor_gerbil(imps.gerbil, imp = 1)
```

```
#Print a summary
cors.gerbil
```

---

gerbil *General Efficient Regression-Based Imputation with Latent processes*

Description

Coherent multiple imputation of general multivariate data as implemented through the GERBIL algorithm described by Robbins (2020). The algorithm is

- **coherent** in that imputations are sampled from a valid joint distribution, ensuring MCMC convergence;
- **general** in that data of general structure (binary, categorical, continuous, etc.) may be allowed;
- **efficient** in that computational performance is optimized using the SWEEP operator for both modeling and sampling;
- **regression-based** in that the joint distribution is built through a sequence of conditional regression models;
- **latent** in that a latent multivariate normal process underpins all variables; and
- **flexible** in that the user may specify which dependencies are enabled within the conditional models.

Usage

```
gerbil(
  dat,
  m = 1,
  mcmciter = 25,
  predMat = NULL,
  type = NULL,
  visitSeq = NULL,
  ords = NULL,
  semi = NULL,
  bincat = NULL,
  cont.meth = "EMP",
  num.cat = 12,
  r = 5,
  verbose = TRUE,
  n.cores = NULL,
  cl.type = NULL,
  mass = rep(0, length(semi)),
  ineligible = NULL,
  trace = TRUE,
  seed = NULL,
  fully.syn = FALSE
)
```

Arguments

dat	The dataset that is to be imputed. Missing values must be coded with NA.
m	The number of multiply imputed datasets to be created. By default, m = 1.
mcmciter	The number of iterations of Markov chain Monte Carlo that will be used to create each imputed dataset. By default, m = 25.
predMat	A numeric matrix of ncol(dat) columns and no more than nrow(dat) rows, containing 0/1 data specifying the set of predictors to be used for each target row. Each row corresponds to a variable. A value of 1 means that the column variable is used as a predictor for the variable in the target row. By default, predMat is a square matrix of ncol(dat) rows and columns with 1's below the diagonal and 0's on and above the diagonal. Any non-zero value on or above the diagonal will be set to zero.
type	A named vector that gives the type of each variable contained in dat. Possible types include 'binary', 'categorical', 'ordinal', 'semicont' (semi-continuous), and 'continuous'. The vector type should be named where the names indicate the corresponding column of dat. Types for variables not listed in type will be determined by default, in which case a variable with no more than num.cat possible values will be set as binary/categorical and is set as continuous otherwise.
visitSeq	A vector of variable names that has (at least) contains all names of each column of dat that has missing values. Within the I-Step and P-Step of gerbil, the variables will be modeled and imputed in the sequence given by visitSeq. If visitSeq = TRUE, visitSeq is reset as being equal to the columns of dat ordered from least to most missingness. If visitSeq = NULL (the default) or visitSeq = FALSE variables are ordered in accordance with the order of the rows of predMat or (if unavailable) the order in which they appear in the dat.
ords	A character string giving a set of the column names of dat that indicate which variables are to be treated as ordinal. Elements of ords are overridden by any conflicting information in type. By default, ords = NULL.
semi	A character string giving a set of the column names of dat that indicate which variables are to be treated as semi-continuous. Elements of semi are overridden by any conflicting information in type. By default, semi = NULL.
bincat	A character string giving a set of the column names of dat that indicate which variables are to be treated as binary or unordered categorical. Elements of bincat are overridden by any conflicting information in type. By default, bincat = NULL.
cont.meth	The type of marginal transformation used for continuous variables. Set to "EMP" by default for the empirical distribution transformation of Robbins (2014). The current version also includes an option for no transformation (cont.meth = "none"). Other transformation types will be available in future versions of gerbil. .
num.cat	Any variable that does not have a type specified by any of the other parameters will be treated as categorical if it takes on no more than num.cat possible values and as continuous if it takes on more than num.cat possible values. By default, num.cat = 12.

r	The number of pairwise completely observed cases that must be available for any pair of variables to have dependencies enabled within the conditional models for imputation. By default, r = 5.
verbose	If TRUE (the default), history is printed on console. Use verbose = FALSE for silent computation.
n.cores	The number of CPU cores to use for parallelization. If n.cores is not specified by the user, it is guessed using the detectCores function in the parallel package. If TRUE (the default), it is set as detectCores(). If NULL, it is set as floor((detectCores()+1)/2). If FALSE, it is set as 1, in which case parallelization is not invoked. Note that the documentation for detectCores makes clear that it is not failsafe and could return a spurious number of available cores. By default, n.cores is set as floor((n + 1)/2), where n is the number of available clusters.
cl.type	The cluster type that is passed into the makeCluster() function in the parallel package. Defaults to 'PSOCK'.
mass	A named vector of the same length as the number of semi-continuous variables in dat that gives the location (value) of the point mass for each such variable. The point of mass for each semicontinuous variable is set to zero by default.
ineligible	Either a scalar or a matrix that is used to determined which values are to be considered missing but ineligible for imputation. Such values will be imputed internally within gerbil to ensure a coherent imputation model but will be reset as missing after imputations have been created. If ineligible is a scalar, all data points that take on the respective value will be considered missing but ineligible for imputation. If ineligible is a matrix (with the same number of rows as dat and column names that overlap with dat), entries of TRUE or 1 in ineligible indicate values that are missing but ineligible for imputation. If ineligible = NULL (the default), all missing values will be considered eligible for imputation.
trace	A logical that, if TRUE, implies that means and variances of variables are tracked across iterations. Set to FALSE to save computation time. However, trace plots and R hat statistics are disabled for gerbil objects created with trace = FALSE. Defaults to TRUE.
seed	An integer that, when specified, is used to set the random number generator via set.seed().
fully.syn	A logical that, if TRUE, implies that a fully synthetic dataset will be created (although variables without missingness are not altered).

Details

gerbil is designed to handle the following classes of variables:

- 'continuous': Variables are transformed to be (nearly) standard normal prior to imputation. The default transformation method is based on empirical distributions (see Robbins, 2014) and ensures that imputed values of a variable are sampled from the observed values of that variable.
- 'binary': Dichotomous variables are handled through probit-type models in that they are underpinned by a unit-variance normally distributed random variable.

- 'categorical': Unordered categorical variables are handled by creating nested binary variables that underpin the categorical data. Missingness is artificially imposed in the nested variables in order to ensure conditional independence between them. See Robbins (2020) for details.
- 'ordinal': Ordered categorical variables (ordinal) are handled through a probit-type model in that a latent normal distribution is assumed to underpin the ordinal observations. See Robbins (2020) for details.
- 'semicont': Mixed discrete/continuous (semi-continuous) variables are assumed to observe a mass at a specific value (most often zero) and are continuous otherwise. A binary variable is created that indicates whether the semi-continuous variable takes on the point-mass value; the continuous portion is set as missing when the observed semi-continuous variable takes on the value at the point-mass. See Robbins et al. (2013) for details.

The parameter type allows the user to specify the class for each variable. Routines are in place to establish the class by default for variables not stated in type. Note that it is not currently possible for a variable to be assigned a class of semi-continuous by default.

gerbil uses a joint modeling approach to imputation that builds a joint model using a sequence of conditional models, as outlined in Robbins et al. (2013). This approach differs from fully conditional specification in that the regression model for any given variable is only allowed to depend upon variables that preceed it in an index ordering. The order is established by the parameter visitSeq. gerbil contains the flexibility to allow its user to establish which of the permissible dependencies are enabled within the conditional models. Enabled dependencies are stated within the parameter predMat. Note that the data matrix used for imputation is an expanded version of the data that are fed into the algorithm (variables are created that underpin unordered categorical and semi-continuous variables). Note also that conditional dependencies between the nested binary variables of a single undordered categorical variables or the discrete and continuous portions of a semi-continuous variable are not permitted.

The output of gerbil is an object of class gerbil which is a list that contains the imputed datasets (imputed), missingness indicators (missing and missing.latent), summary information (summary), output used for MCMC convergence diagostics (chainSeq and R.hat), and modeling summaries (visitSeq.initial, visitSeq.final, predMat.initial, predMat.final, drops, and forms). Some output regarding convergence diagnostics and modeling regards the expanded dataset used for imputation (the expanded dataset includes binary indicators for unordered categorical and semi-continuous variables). Note that the nested binary variables corresponding to an unordered categorical variable X with categories labeled a, b, c, etc., are named X.a, X.b, X.c, and so forth in the expanded dataset. Likewise, the binary variable indicating the point mass of a semi-continuous variable Y is named Y.B in the expanded dataset, and the positive portion (with missingness imposed) is left as being named Y.

gerbil automatically checks each regression model for perfect collinearities and reduces the model as needed. Variables that have been dropped from a given model are listed in the element named 'drops' in a gerbil object.

Value

gerbil() returns an object the class gerbil that contains the following slots:

imputed A list of length m that contains the imputed datasets.

missing A matrix 0s, 1s, 2s, and 4s of the same dimension as dat that indicates which values were observed or missing. A 0 indicates a fully observed value, a 1 indicates a missing value that was imputed, and a 4 indicates a missing value that was ineligible for imputation.

summary A matrix with ncol(dat) number of rows that contains summary information, including the type of each variable and missingness rates. Note that for continuous variables, the type listed indicates the method of transformation used.

chainSeq A list of six elements. Each element is a matrix with mcmciter columns and up to ncol(dat) rows. Objects means.all and means.mis give the variables means of data process across iterations of MCMC when all observations are incorporated and when only imputed values are incorporated, respectively. (Means of continuous variables are given on the transformed scale.) Similar objects are provided to track variances of variables. Variables are listed in the order provided by the gerbil object visitSeq.latent. Variables reported in this output are those contained in the dataset that has been expanded to include binary indicators for categorical and semi-continuous variables.

R.hat The value of the R hat statistics of Gelman and Rubin (1992) for the means and variances of each variable. The R hat statistic is also provided for mean of binary variables. Variables include those contained in the expanded dataset and are listed in the order provided by object visitSeq.latent. Only calculated if m > 2 and mcmciter >= 4.

missing.latent A matrix of the same dimensions as the expanded dataset, but used to indicate missingness in the expanded dataset. In this matrix, 0s indicate fully observed values, 1s indicate fully missing values, 3s indicate values that have imposed missingness (for binary indicators corresponding to categorical or semi-continuous variables), and 4 indicates a missing value that is ineligible for imputation (as determined by the input 'ineligible')..

visitSeq.initial A vector of variable names giving the sequential ordering of variables that is used for imputation prior to expanding the dataset include nested binary and point-mass indicators. Variables without missing values are excluded.

visitSeq.final A vector of variable names giving the sequential ordering of variables in the expanded dataset that is used for imputation. Variables without missing values are excluded.

predMat.initial A matrix of ones and zeros indicating the dependencies enabled in the conditional models used for imputation. This matrix is determined from the input 'predMat'. Rows corresponding to variables with no missing values are removed.

predMat.final A matrix of ones and zeros indicating the dependencies enabled in the conditional models used for imputation. This is of a similar format to the input 'predMat' but pertains to the expanded dataset. Rows corresponding to variables with no missing values are removed.

drops A list of length equal to the number of variables in the expanded dataset that have missing values. Elements of the list indicate which variables were dropped from the conditional model for the corresponding variable due to either insufficient pairwise complete observations (see the input 'r') or perfect collinearities.

forms A list of length equal to the number of variables in the expanded dataset that have missing values. Elements of the list indicate the regression formula used for imputation of the respective variable.

mass.final The final version of the input parameter mass.

ineligibles A logical matrix with the same number of rows and columns as dat that indicates which elements are considered missing but ineligible for imputation.

nams.out A vector used to link column names in the expanded data to corresponding names in the original data.

References

Gelman, A., & Rubin, D. B. (1992). Inference from iterative simulation using multiple sequences. *Statistical Science*, 7(4), 457-472.

Robbins, M. W. (2014). The Utility of Nonparametric Transformations for Imputation of Survey Data. *Journal of Official Statistics*, 30(4), 675-700.

Robbins, M. W. (2020). A flexible and efficient algorithm for joint imputation of general data. arXiv preprint arXiv:2008.02243.

Robbins, M. W., Ghosh, S. K., & Habiger, J. D. (2013). Imputation in high-dimensional economic data as applied to the Agricultural Resource Management Survey. *Journal of the American Statistical Association*, 108(501), 81-95.

Examples

```
#Load the India Human Development Survey-II dataset
data(ihd_mcar)

# Gerbil without types specified
imps.gerbil <- gerbil(ihd_mcar, m = 1, mcmciter = 10)

# Gerbil with types specified (method #1)
types.gerbil <- c(
       sex = "binary", age = "continuous",
       marital_status = "binary", job_field = "categorical",
       farm_labour_days = "semicont", own_livestock = "binary",
       education_level = "ordinal", income = "continuous")
imps.gerbil <- gerbil(ihd_mcar, m = 1, type = types.gerbil)

# Gerbil with types specified (method #2)
imps.gerbil <- gerbil(ihd_mcar, m = 1, ords = "education_level", semi = "farm_labour_days",
       bincat = c("sex", "marital_status", "job_field", "own_livestock"))

# Gerbil with types specified (method #3)
types.gerbil <- c("binary", "continuous", "binary", "categorical", "semicont",
       "binary", "ordinal", "continuous")
imps.gerbil <- gerbil(ihd_mcar, m = 1, type = types.gerbil)

# Variables of class factor are treated as binary/categorical by default
ihd.fac <- ihd_mcar
ihd.fac$sex <- factor(ihd_mcar$sex)
ihd.fac$marital_status <- factor(ihd_mcar$marital_status)
ihd.fac$job_field <- factor(ihd_mcar$job_field)
ihd.fac$own_livestock <- factor(ihd_mcar$own_livestock)
ihd.fac$education_level <- ordered(ihd_mcar$education_level)
imps.gerbil <- gerbil(ihd.fac, m = 1)

# Univariate plotting of one variable
plot(imps.gerbil, type = 1, y = "job_field")

# gerbil with predMat specified (method #1)
predMat <- matrix(c(1, 0, 0, 1), 2, 2)
```

```
dimnames(predMat) <- list(c("education_level", "income"), c("sex", "job_field"))
imps.gerbil <- gerbil(ihd_mcar, m = 1, type = types.gerbil, predMat = predMat)

# gerbil with predMat specified (method #2)
predMat <- rbind(
       c(0, 0, 0, 0, 0, 0, 0, 0),
       c(1, 0, 0, 0, 0, 0, 0, 0),
       c(1, 1, 0, 0, 0, 0, 0, 0),
       c(1, 1, 1, 0, 0, 0, 0, 0),
       c(1, 1, 1, 1, 0, 0, 0, 0),
       c(1, 1, 1, 1, 1, 0, 0, 0),
       c(1, 1, 1, 0, 1, 1, 0, 0),
       c(0, 1, 1, 1, 1, 1, 1, 0)
       )
imps.gerbil <- gerbil(ihd_mcar, type = types.gerbil, predMat = predMat)

# Multiple imputation with more iterations
imps.gerbil.5 <- gerbil(ihd_mcar, m = 5, mcmciter = 100, ords = "education_level",
       semi = "farm_labour_days", bincat = "job_field", n.cores = 1)

plot(imps.gerbil.5, type = 1, y = "job_field", imp = 1:5)

# Extract the first imputed dataset
imputed.gerb <- imputed(imps.gerbil.5, imp = 1)

# Write all imputed datasets to an Excel file
write.gerbil(imps.gerbil.5, file = file.path(tempdir(), "gerbil_example.xlsx"), imp = 1:5)


## Not run:
if(requireNamespace('mice')){
# Impute using mice for comparison

types.mice <- c("logreg", "pmm", "logreg", "polyreg", "pmm", "logreg", "pmm", "pmm")
imps.mice <- mice(ihd.fac, m = 1, method = types.mice, maxit = 100)

imps.mice1 <- mice(ihd.fac, m = 1, method = "pmm", maxit = 100)

imps.gerbil <- gerbil(ihd_mcar, m = 1, mcmciter = 100, ords = "education_level",
    semi = "farm_labour_days", bincat = "job_field")

# Compare the performance of mice and gerbil

# Replace some gerbil datasets with mice datasets
imps.gerbil.m <- imps.gerbil.5
imps.gerbil.m$imputed[[2]] <- complete(imps.mice, action = 1)
imps.gerbil.m$imputed[[3]] <- complete(imps.mice1, action = 1)

# Perform comparative correaltion analysis
cor_gerbil(imps.gerbil.m, imp = 1, log = "income")
cor_gerbil(imps.gerbil.m, imp = 2, log = "income")
cor_gerbil(imps.gerbil.m, imp = 3, log = "income")
```

```
# Perform comparative univariate goodness-of-fit testing
gof_gerbil(imps.gerbil.m, type = 1, imp = 1)
gof_gerbil(imps.gerbil.m, type = 1, imp = 2)
gof_gerbil(imps.gerbil.m, type = 1, imp = 3)

# Perform comparative bivariate goodness-of-fit testing
gof_gerbil(imps.gerbil.m, type = 2, imp = 1)
gof_gerbil(imps.gerbil.m, type = 2, imp = 2)
gof_gerbil(imps.gerbil.m, type = 2, imp = 3)

# Produce univariate plots for comparisons
plot(imps.gerbil.m, type = 1, file = file.path(tempdir(), "gerbil_vs_mice_univariate.pdf"),
     imp = c(1, 2, 3), log = "income", lty = c(1, 2, 4, 5), col = c("blue4", "brown2",
    "green3", "orange2"), legend = c("Observed", "gerbil", "mice: logistic", "mice: pmm"))

### Produce bivariate plots for comparisons
plot(imps.gerbil.m, type = 2, file = file.path(tempdir(), "gerbil_vs_mice_bivariate.pdf"),
    imp = c(1, 2, 3), log = "income", lty = c(1, 2, 4, 5), col = c("blue4", "brown2",
    "green3", "orange2"), pch = c(1, 3, 4, 5), legend = c("Observed", "gerbil",
    "mice: logistic", "mice: pmm"))

}

## End(Not run)
```

`gof_gerbil` *Goodness-of-fit testing for* `gerbil` *objects*

Description

Using a `gerbil` object as an input, this function performs univariate and bivariate goodness-of-fit tests to compare distributions of imputed and observed values.

Usage

```
gof_gerbil(
  x,
  y = NULL,
  type = 1,
  imp = 1,
  breaks = NULL,
  method = c("chi-squared", "fisher", "G"),
  ks = FALSE,
  partial = "imputed",
  ...
)
```

Arguments

x	A gerbil object containing the imputed data.
y	A vector listing the column names of the imputed data for which tests should be run. See details. By default, y contains all columns of the data that required imputation.
type	A scalar used to specify the type of tests that will be performed. Options include univariate (marginal) tests (type = 1) and bivariate tests (type = 2). See details. Defaults to type = 1.
imp	A scalar or vector indicating which of the multiply imputed datasets should be used for testing. Defaults to imp = 1.
breaks	Used to determine the cut-points for binning of continuous variables into categories. Ideally, breaks is a named list, where the list names are the names of the continuous variables. Each element of the list can be a vector giving the respective cutpoints or a scalar which is used to indicate the number of bins (in which case cutpoints are determined from percentiles in order to yield bins of approximately equal size). If breaks is a scalar or a vector (and not a list), the binning strategy indicated by breaks is applied to each variable in accordance with the description above. Defaults to breaks = 4.
method	The type of test that is used to compare contingency tables. Options include 'chi-squared' for chi-squared testing (the default), 'fisher' for Fisher's exact test, and 'G' for a G-test.
ks	If TRUE, a Kolmogorov-Smirnov test is used when for univariate comparisons with continuous variables. This functionality is not enabled for bivariate testing. Defaults to FALSE.
partial	Indicates how partially imputed pairs are handled in bivariate testing. If 'imputed', cases with at least one missing variable in a pair are considered imputed. Otherwise (partial = 'observed'), only cases with both variables in the pair missing are considered imputed.
...	Arguments to be passed to methods.

Details

Goodness of fit is determined using contingency tables of counts across categories of the corresponding variable(s). For univariate testing (type = 1), a one-way table is calculated for observed cases and compared to an analogous table for imputed cases, whereas for bivariate testing (type = 2), two-way tables are calculated. Continuous variables are binned according to cut-points defined using the parameter breaks. Tests are performed using one of three methods (determined from the parameter method): 1) Chi-squared (the default); 2) Fisher's exact; and 3) A G-test. G-testing is implemented via the function GTest() from the DescTools package. Note that for univariate testing of continuous variables, a Kolmogorov-Smirnov test may be performed instead by setting ks = TRUE.

The only required input is a parameter x which is a gerbil object.

Note that univariate differences between observed and imputed data may be explained by the missingness mechanism and are not necessarily indicative of poor imputations. Note also that most imputation methods like gerbil (and mice and related methods) are not designed to capture complete bivariate distributions. As such, the bivariate tests may be likely to return small p-values.

Value

gof_gerbil() returns an object of the class gof_gerbil that has following slots:

Stats A vector (when type = 1) or matrix (when type = 2) giving the value of the test statistic (or coefficient) for the corresponding variable (or variable pair).

p.values A vector (when type = 1) or matrix (when type = 2) giving the value of the p-value for the test applied to the corresponding variable (or variable pair).

Test A vector (when type = 1) or matrix (when type = 2) indicating the type of test applied to the corresponding variable (or variable pair).

Breaks A list giving the cutpoints used for binning each continuous or semi-continuous variable.

Examples

```
#Load the India Human Development Survey-II dataset
data(ihd_mcar)

imps.gerbil <- gerbil(ihd_mcar, m = 1, mcmciter = 200, ords = "education_level",
     semi = "farm_labour_days", bincat = c("sex", "marital_status", "job_field", "own_livestock"))

#Run univariate tests
tests.gerbil.uni <- gof_gerbil(imps.gerbil, imp = 1, type = 1)

#Print a summary
tests.gerbil.uni

#Run bivariate tests
tests.gerbil.bi <- gof_gerbil(imps.gerbil, imp = 1, type = 2)

#Print a summary
tests.gerbil.bi
```

---

ihd *Example data from the India Human Development Survey*

Description

This dataset is a subset from the India Human Development survey. This dataset is included in the package only for demonstration purposes and should not be used for other purposes.

Usage

```
ihd
```

Format

A data frame with 42155 rows and 8 variables:

sex 0 = individual is male; 1 = individual is female

age Age of the individual, between 0 & 99

marital_status Individual's marital status. 0 = Unmarried; 1 = Married

job_field Refer's to the field of the individual's profession or job status (i.e. agricultural worker; small business owner; student; unemployed; etc.

farm_labour_days Number of days a year the individual worked on a farm. Can take on the value zero

own_livestock 0 = Individual does not own livestock. 1 = Individual does own livestock

education_level Years of schooling attained by the individual. Censored for values above 16.

income Household's income, in rupees. Value can be negative

Source

Desai, Sonalde, and Vanneman, Reeve. India Human Development Survey-II (IHDS-II), 2011-12. Inter-university Consortium for Political and Social Research [distributor], 2018-08-08. doi:10.3886/ICPSR36151.v6

---

ihd_mar *Missing at Random example data from the India Human Development Survey*

Description

This dataset is a subset from the India Human Development survey. This dataset is included in the package only for demonstration purposes and should not be used for other purposes.

Usage

```
ihd_mar
```

Format

A data frame with 42155 rows and 8 variables:

sex 1 = individual is male; 2 = individual is female

age Age of the individual, between 0 & 99

marital_status Individual's marital status. 0 = married, absent spouse, 1 = Married, 2 = Unmarried, 3 = Widowed, 4 = Divorced/Separated, 5 = married, no gauna

job_field Refer's to the field of the individual's profession or job status (i.e. agricultural worker; small business owner; student; unemployed; etc.

farm_labour_days Number of days a year the individual worked on a farm. Can take on the value zero

own_livestock 0 = Individual does not own livestock. 1 = Individual does own livestock

education_level Years of schooling attained by the individual. Censored for values above 16.

income Household's income, in rupees. Value can be negative

Source

Desai, Sonalde, and Vanneman, Reeve. India Human Development Survey-II (IHDS-II), 2011-12. Inter-university Consortium for Political and Social Research [distributor], 2018-08-08. doi:10.3886/ICPSR36151.v6

`ihd_mcar` *Missing Completely at Random example data from the India Human Development Survey*

Description

This dataset is a subset from the India Human Development survey. This dataset is included in the package only for demonstration purposes and should not be used for other purposes.

Usage

```
ihd_mcar
```

Format

A data frame with 42155 rows and 8 variables:

sex 1 = individual is male; 2 = individual is female

age Age of the individual, between 0 & 99

marital_status Individual's marital status. 0 = married, absent spouse, 1 = Married, 2 = Unmarried, 3 = Widowed, 4 = Divorced/Separated, 5 = married, no gauna

job_field Refer's to the field of the individual's profession or job status (i.e. agricultural worker; small business owner; student; unemployed; etc.

farm_labour_days Number of days a year the individual worked on a farm. Can take on the value zero

own_livestock 0 = Individual does not own livestock. 1 = Individual does own livestock

education_level Years of schooling attained by the individual. Censored for values above 16.

income Household's income, in rupees. Value can be negative

Source

Desai, Sonalde, and Vanneman, Reeve. India Human Development Survey-II (IHDS-II), 2011-12. Inter-university Consortium for Political and Social Research [distributor], 2018-08-08. doi:10.3886/ICPSR36151.v6

imputed *Extracting imputed datasets from gerbil objects*

Description

Using a gerbil object as an input, this function returns imputed datasets.

Usage

```
imputed(gerb, imp = 1)
```

Arguments

gerb	A gerbil object containing the imputed data.
imp	The imputed datasets which are to be returned (defaults to imp = 1). Letting m indicate the number of imputed datasets contained in gerb, imp should be a subset of 1:m. If imp is a scalar, a single imputed dataset is returned. If imp is a vector, then the individual datasets are stacked on top of each other and returned.

Details

The function either return a single imputed dataset (if imp is a scalar) or a tall dataset if (if imp is a vector) with the individual datsets stacked on top of each other.

Value

imputed() returns a data frame or matrix. If imp has multiple elements, columns are added to indicate the imputation number and the case ID.

Examples

```
#Load the India Human Development Survey-II dataset
data(ihd_mcar)

# Create a gerbil object
imps.gerbil <- gerbil(ihd_mcar, m = 5, ords = "education_level", semi = "farm_labour_days",
       bincat = "job_field", n.cores = 1)

# Return a single imputed datasets
imp.gerb <- imputed(imps.gerbil, imp = 2)

# Return multiple (stacked) datasets
imp.gerb <- imputed(imps.gerbil, imp = 1:5)
```

plot.gerbil *Plotting for gerbil objects*

Description

Using a gerbil object as an input, this function gives diagnostic plots for selected variables

Usage

```
## S3 method for class 'gerbil'
plot(
  x,
  y = NULL,
  type = "Univariate",
  imp = 1,
  col = NULL,
  lty = NULL,
  lwd = NULL,
  pch = NULL,
  log = NULL,
  legend = NULL,
  legend.loc = "topright",
  mfrow = c(3, 2),
  trace.type = "Mean",
  file = NULL,
  sep = FALSE,
  height = NULL,
  width = NULL,
  partial = "imputed",
  ...
)
```

Arguments

x	A gerbil object containing the imputed data.
y	A vector listing the column names of the imputed data for which plots should be created. See details. By default, y contains all columns of the data that required imputation.
type	A scalar used to specify the type of plots that will be created. Options include univariate (marginal) plots (type = 1), bivariate plots (type = 2), and trace plots (type = 3). See details. Defaults to type = 1.
imp	A scalar or vector indicating which of the multiply imputed datasets should be used for plotting. Defaults to imp = 1. Setting imp = TRUE will include all imputed datasets.

col	The color used for plotting – should be a vector of length equal to imp + 1. The first element references plotting of observed data, and remaining elements reference plotting of imputed data.
lty	The line type used for plotting imputed values with trace lines or density plots – should be a vector of length equal to imp + 1. The first element references plotting of observed data, and remaining elements reference plotting of imputed data.
lwd	The line width used for density and trace line plotting – should be a vector of length equal to imp + 1. The first element references plotting of observed data, and remaining elements reference plotting of imputed data.
pch	A length-2 vector that indicates the plotting symbol to be used for imputed and observed values in scatter and lattice plots.
log	A character vector that includes names of variables of which a log transformation is to be taken prior to plotting.
legend	A character or expression vector to appear in the legend. If FALSE or 'n', no legend is created. Defaults to c("Observed", "Imputed", ...).
legend.loc	The location of the legend in the plots.
mfrow	The layout of plots across a single page when there are to be multiple plots per page (as is the case when file is non-NULL and sep = FALSE).
trace.type	The type of trace plot to be created (only valid when type = 3). See details. Defaults to trace.type = 1.
file	A character string giving the name of file that will be created in the home directory containing plots. The name should have a .pdf or .png extension. If NULL (the default), no file is created.
sep	If sep = TRUE, separate plots will be generated for each outcome. Applicable only if plots are saved to file (plot.file is non-NULL). To change display of plots produced as output, use par.
height	The height of the graphics region (in inches) when a pdf is created.
width	The width of the graphics region (in inches) when a pdf is created.
partial	Indicates how partially imputed pairs are handled in bivariate plotting. If 'imputed', cases with at least one missing variable in a pair are considered imputed. Otherwise (partial = 'observed'), only cases with both variables in the pair missing are considered imputed.
...	Arguments to be passed to methods, such as plot.

Details

Three types of plots may be produced: 1) Univariate (produced by setting type = 1): Compares the marginal distribution of observed and imputed values of a given variable. Density plots are produced for continuous variables, and bar plots are given for binary, categorical, and ordinal variables. For semi-continuous variables, two plots are constructed: a) a bar plot for the binary portion of the variable and 2) a density plot for the continuous portion. 2) Bivariate (produced by setting type = 2): Compares the bivariate distributions of observed and imputed values of two variables. Scatter plots are produced if both variables are continuous or semi-continuous, box plots are produced if

one variable is continuous or semi-continuous and the other is not, and a lattice plot is produced if neither variable is continuous or semi-continuous. For bivariate plots, imputed observations are those that have one or more of the values of the pair missing within the original dataset. 3) Trace lines (produced by setting `type = 3`): Plots a pre-specified parameter across iterations of MCMC in order to examine convergence for a given variable. Parameters that may be plotted include means (`trace.type = 1`) and variances (`trace.type = 2`).

Multiple plots may be created, as determined by the variable names listed in the parameter `y`. For univariate and trace plots, one plot is created for each variable listed in `y`. For bivariate plotting, one plot is created for each combination of two elements within the vector `y` (as such, `y` must have a length of at least two in this case). For trace plotting, elements of `y` should correspond to column names in the dataset that has been expanded to include binary indicators for categorical and semi-continuous variables. If multiple plots are to be created, it is recommended to specify a file for output using the parameter `file`, in which case separate files will be created for each plot (if `sep = TRUE`) or all plots will be written to the same file (if `sep = FALSE`).

The only required input is a parameter `x` which is a `gerbil` object.

Value

No returned value, but instead plots are generated in the workspace or written to a specified directory.

Examples

```
#Load the India Human Development Survey-II dataset
data(ihd_mcar)

# Create a gerbil object

imps.gerbil <- gerbil(ihd_mcar, m = 1, ords = "education_level", semi = "farm_labour_days",
       bincat = "job_field")

# Univariate plotting of all variables to a file
plot(imps.gerbil, type = 1, file = file.path(tempdir(), "gerbil_univariate.pdf"))

# Bivariate plotting of all variables to a file
plot(imps.gerbil, type = 2, file = file.path(tempdir(), "gerbil_bivariate.pdf"))

# Trace plotting of all variables to a file
plot(imps.gerbil, type = 3, file = file.path(tempdir(), "gerbil_ts.pdf"))

# Univariate plotting of one variable (not to a file)
plot(imps.gerbil, type = 1, y = "job_field")

# Bivariate plotting of one pair of variables (not to a file)
plot(imps.gerbil, type = 2, y = c("job_field", "income"))

# Bivariate plotting of one pair of variables (not to a file) with income logged
plot(imps.gerbil, type = 2, y = c("job_field", "income"), log = "income")
```

`print.cor_gerbil` *Prints a* `cor_gerbil` *object. Printed output includes the average difference of correlations, as well as summaries of the test statistics based on Fisher's z and their p-values.*

Description

Prints a `cor_gerbil` object. Printed output includes the average difference of correlations, as well as summaries of the test statistics based on Fisher's z and their p-values.

Usage

```
## S3 method for class 'cor_gerbil'
print(x, ...)
```

Arguments

`x`	object of `cor_gerbil` class
`...`	additional parameters to be passed down to inner functions.

Value

The functions `print.cor_gerbil` and `summary.cor_gerbil` display information about the `cor_gerbil` object. The output displayed includes: 1) the average absolute difference in correlation between observed and imputed cases across all relevant variable pairs, 2) the average value of the test statistic based on Fisher's z across all variable pairs, 3) the largest test statistic observed across any variable pair, and 4) the portion of p-values for the test based on Fisher's z that are less than 0.05.

`print.gerbil` *Prints a* `gerbil` *object. Printed output includes a variable-by-variable summary of variable types and missingness rates. The implemented predictor matrix is also provided.*

Description

Prints a `gerbil` object. Printed output includes a variable-by-variable summary of variable types and missingness rates. The implemented predictor matrix is also provided.

Usage

```
## S3 method for class 'gerbil'
print(x, ...)
```

Arguments

x	object of gerbil class
...	additional parameters to be passed down to inner functions.

Value

The functions print.gerbil and summary.gerbil display information about the gerbil object. Primarily, the variable type and missingness rate are displayed for each variable. The predictor matrix is also provided.

---

print.gof_gerbil *Prints a* gof_gerbil *object. Printed output pertains to the goodness-of-fit tests that are applied in order to compare the distribution between observed and imputed cases for relevant variables or variable pairs.*

Description

Prints a gof_gerbil object. Printed output pertains to the goodness-of-fit tests that are applied in order to compare the distribution between observed and imputed cases for relevant variables or variable pairs.

Usage

```
## S3 method for class 'gof_gerbil'
print(x, ...)
```

Arguments

x	object of gof_gerbil class
...	additional parameters to be passed down to inner functions.

Value

The functions print.gof_gerbil and summary.gof_gerbil display information about the cor_gerbil object. The output displayed includes: 1) the average test statistic value across all variables or variable pairs contained in the object, 2) the average p-value of all goodness-of-fit tests contained within the object, and 3) the number of tests that yieled a p-value of less than 0.05.

summary.cor_gerbil *Summarises a* gerbil *object. Printed output includes a variable-by-variable summary of variable types and missingness rates. The implemented predictor matrix is also provided.*

Description

Summarises a gerbil object. Printed output includes a variable-by-variable summary of variable types and missingness rates. The implemented predictor matrix is also provided.

Usage

```
## S3 method for class 'cor_gerbil'
summary(object, ...)
```

Arguments

object	An object of cor_gerbil class
...	additional parameters to be passed down to inner functions.

Value

The functions print.cor_gerbil and summary.cor_gerbil display information about the cor_gerbil object. The output displayed includes: 1) the average absolute difference in correlation between observed and imputed cases across all relevant variable pairs, 2) the average value of the test statistic based on Fisher's z across all variable pairs, 3) the largest test statistic observed across any variable pair, and 4) the portion of p-values for the test based on Fisher's z that are less than 0.05.

summary.gerbil *Summarises a* gerbil *object. Printed output includes a variable-by-variable summary of variable types and missingness rates. The implemented predictor matrix is also provided.*

Description

Summarises a gerbil object. Printed output includes a variable-by-variable summary of variable types and missingness rates. The implemented predictor matrix is also provided.

Usage

```
## S3 method for class 'gerbil'
summary(object, ...)
```

Arguments

object	An object of gerbil class
...	Additional parameters to be passed down to inner functions.

Value

The functions print.gerbil and summary.gerbil display information about the gerbil object. Primarily, the variable type and missingness rate are displayed for each variable. The predictor matrix is also provided.

---

summary.gof_gerbil *Summarises a* gerbil *object. Printed output pertains to the goodness-of-fit tests that are applied in order to compare the distribution between observed and imputed cases for relevant variables or variable pairs.*

Description

Summarises a gerbil object. Printed output pertains to the goodness-of-fit tests that are applied in order to compare the distribution between observed and imputed cases for relevant variables or variable pairs.

Usage

```
## S3 method for class 'gof_gerbil'
summary(object, ...)
```

Arguments

object	An object of gof_gerbil class
...	Additional parameters to be passed down to inner functions.

Value

The functions print.gof_gerbil and summary.gof_gerbil display information about the cor_gerbil object. The output displayed includes: 1) the average test statistic value across all variables or variable pairs contained in the object, 2) the average p-value of all goodness-of-fit tests contained within the object, and 3) the number of tests that yieled a p-value of less than 0.05.

`write.gerbil` *Write imputed datasets from gerbil objects to a file or files*

Description

Using a `gerbil` object as an input, this function writes imputed datasets to an output file.

Usage

```
write.gerbil(gerb, file = NULL, imp = NULL, tall = FALSE, row.names = FALSE)
```

Arguments

`gerb`	A `gerbil` object containing the imputed data.
`file`	The name of the file to which the imputed datasets are to be written. Which type of file (.xlsx or .csv) is created depends upon the extension of the parameter `file`.
`imp`	The imputed datasets which are to be written. Can be a scalar or, if multiple imputed datasets are to be written, a vector. All elements of `imp` should be integers greater than 0 but no greater than `m`, which is the number of imputed datasets in `gerb`. `imp` defaults to `1:m`.
`tall`	A logical expression indicating whether the datasets are to be written in a tall (stacked) format or written separately. When writing to an XLSX file with `tall = FALSE`, one tab is created for each imputed dataset. When writing to a CSV file with `tall = FALSE`, one file is created for each imputed dataset (in this case, several file names will be created from the base string given by `file`).
`row.names`	A logical value indicating whether the row names of the datasets are to be written.

Details

The function writes imputed datasets to either an Excel (.xlsx) or a CSV (.csv) file, depending upon the extension of the parameter `file`. No other file types are supported. To write multiple imputed datasets simultaneously, specify `imp` as a vector with length greater than 1. Multiple imputed datasets are either written in a stacked format (if `tall = TRUE`) or written separately (if `tall = FALSE`).

Value

No returned value, but instead a data file is written to a specified directory.

Examples

```
#Load the India Human Development Survey-II dataset
data(ihd_mcar)

# Create a gerbil object
```

```
imps.gerbil <- gerbil(ihd_mcar, m = 5, ords = "education_level", semi = "farm_labour_days",
       bincat = "job_field", n.cores = 1)

# Write all imputed datasets to separate CSV files
write.gerbil(imps.gerbil, file.path(tempdir(), "gerbil_example.csv"), imp = 1:5, tall = FALSE)

# Write all imputed datasets to a single CSV files
write.gerbil(imps.gerbil, file.path(tempdir(), "gerbil_example.csv"), imp = 1:5, tall = TRUE)

# Write all imputed datasets to an XLSX file
write.gerbil(imps.gerbil, file.path(tempdir(), "gerbil_example.xlsx"), imp = 1:5, tall = FALSE)
```

Index