Package ‘modi’

July 23, 2025

Type Package

Title Multivariate Outlier Detection and Imputation for Incomplete Survey Data

Version 0.1.2

Description Algorithms for multivariate outlier detection when missing values occur. Algorithms are based on Mahalanobis distance or data depth. Imputation is based on the multivariate normal model or uses nearest neighbour donors. The algorithms take sample designs, in particular weighting, into account. The methods are described in Bill and Hulliger (2016) <doi:10.17713/ajs.v45i1.86>.

License MIT + file LICENSE

URL `https://github.com/martinSter/modi`

BugReports `https://github.com/martinSter/modi/issues`

Language en-GB

Encoding UTF-8

LazyData true

Depends R (>= 3.5.0)

Imports MASS (>= 7.3-50), norm (>= 1.0-9.5), stats, graphics, utils

RoxygenNote 7.2.3

Suggests knitr, rmarkdown, survey, testthat

VignetteBuilder knitr

NeedsCompilation no

Author Beat Hulliger [aut, cre],
Martin Sterchi [ctb],
Tobias Schoch [ctb]

Maintainer Beat Hulliger <`beat.hulliger@fhnw.ch`>

Repository CRAN

Date/Publication 2023-03-14 09:40:02 UTC

Contents

BEM *BACON-EEM Algorithm for multivariate outlier detection in incomplete multivariate survey data*

Description

BEM starts from a set of uncontaminated data with possible missing values, applies a version of the EM-algorithm to estimate the center and scatter of the good data, then adds (or deletes) observations to the good data which have a Mahalanobis distance below a threshold. This process iterates until the good data remain stable. Observations not among the good data are outliers.

Usage

```
BEM(
  data,
  weights,
  v = 2,
  c0 = 3,
  alpha = 0.01,
  md.type = "m",
  em.steps.start = 10,
  em.steps.loop = 5,
  better.estimation = FALSE,
  monitor = FALSE
)
```

Arguments

`data` a matrix or data frame. As usual, rows are observations and columns are variables.

`weights` a non-negative and non-zero vector of weights for each observation. Its length must equal the number of rows of the data. Default is `rep(1, nrow(data))`.

`v` an integer indicating the distance for the definition of the starting good subset: `v = 1` uses the Mahalanobis distance based on the weighted mean and covariance, `v = 2` uses the Euclidean distance from the componentwise median.

`c0` the size of initial subset is `c0 * ncol(data)`.

`alpha` a small probability indicating the level `(1 - alpha)` of the cutoff quantile for good observations.

`md.type` type of Mahalanobis distance: `"m"` marginal, `"c"` conditional.

`em.steps.start` number of iterations of EM-algorithm for starting good subset.

`em.steps.loop` number of iterations of EM-algorithm for good subset.

`better.estimation` if `better.estimation = TRUE`, then the EM-algorithm for the final good subset iterates `em.steps.start` more.

`monitor` if `TRUE`, verbose output.

Details

The BACON algorithm with `v = 1` is not robust but affine equivariant while `v = 1` is robust but not affine equivariant. The threshold for the (squared) Mahalanobis distances, beyond which an observation is an outlier, is a standardised chisquare quantile at `(1 - alpha)`. For large data sets it may be better to choose `alpha / n` instead. The internal function `EM.normal` is usually called from `BEM`. `EM.normal` is implementing the EM-algorithm in such a way that part of the calculations can be saved to be reused in the `BEM` algorithm. `EM.normal` does not contain the computation of the observed sufficient statistics, they will be computed in the main program of `BEM` and passed as parameters as well as the statistics on the missingness patterns.

Value

`BEM` returns a list whose first component `output` is a sublist with the following components:

`sample.size` Number of observations

`discarded.observations` Number of discarded observations

`number.of.variables` Number of variables

`significance.level` The probability used for the cutpoint, i.e. `alpha`

`initial.basic.subset.size` Size of initial good subset

`final.basic.subset.size` Size of final good subset

`number.of.iterations` Number of iterations of the BACON step

`computation.time` Elapsed computation time

`center` Final estimate of the center

`scatter` Final estimate of the covariance matrix

`cutpoint` The threshold MD-value for the cut-off of outliers

The further components returned by `BEM` are:

`outind` Indicator of outliers

`dist` Final Mahalanobis distances

Note

`BEM` uses an adapted version of the EM-algorithm in function `.EM-normal`.

Author(s)

Beat Hulliger

References

Béguin, C. and Hulliger, B. (2008) The BACON-EEM Algorithm for Multivariate Outlier Detection in Incomplete Survey Data, Survey Methodology, Vol. 34, No. 1, pp. 91-103.

Billor, N., Hadi, A.S. and Vellemann, P.F. (2000). BACON: Blocked Adaptative Computationally-efficient Outlier Nominators. Computational Statistics and Data Analysis, 34(3), 279-298.

Schafer J.L. (2000), Analysis of Incomplete Multivariate Data, Monographs on Statistics and Applied Probability 72, Chapman & Hall.

Examples

```
# Bushfire data set with 20% MCAR
data(bushfirem, bushfire.weights)
bem.res <- BEM(bushfirem, bushfire.weights,
               alpha = (1 - 0.01 / nrow(bushfirem)))
print(bem.res$output)
```

`bushfire` *Bushfire scars.*

Description

The bushfire data set was used by Campbell (1984, 1989) to locate bushfire scars. The dataset contains satellite measurements on five frequency bands, corresponding to each of 38 pixels.

Usage

```
bushfire
```

Format

A data frame with 38 rows and 5 variables.

Details

The data contains an outlying cluster of observations 33 to 38 a second outlier cluster of observations 7 to 11 and a few more isolated outliers, namely observations 12, 13, 31 and 32.

For testing purposes weights are provided: `bushfire.weights <- rep(c(1,2,5), length = nrow(bushfire))`

References

Campbell, N. (1989) Bushfire Mapping using NOAA AVHRR Data. Technical Report. Commonwealth Scientific and Industrial Research Organisation, North Ryde.

Examples

```
data(bushfire)
```

`bushfire.weights` *Weights for Bushfire scars.*

Description

The bushfire data set was used by Campbell (1984, 1989) to locate bushfire scars. The dataset contains satellite measurements on five frequency bands, corresponding to each of 38 pixels.

Usage

```
bushfire.weights
```

Format

A vector of length 38.

Details

For testing purposes, `bushfire.weights` provides artificial weights created according to: `bushfire.weights <- rep(c(1,2,5), length = nrow(bushfire))`

References

Campbell, N. (1989) Bushfire Mapping using NOAA AVHRR Data. Technical Report. Commonwealth Scientific and Industrial Research Organisation, North Ryde.

Examples

```
data(bushfire.weights)
```

bushfirem *Bushfire scars with missing data.*

Description

The bushfire data set was used by Campbell (1984, 1989) to locate bushfire scars. The dataset contains satellite measurements on five frequency bands, corresponding to each of 38 pixels. However, this dataset contains missing values.

Usage

```
bushfirem
```

Format

A data frame with 38 rows and 5 variables.

Details

The data contains an outlying cluster of observations 33 to 38 a second outlier cluster of observations 7 to 11 and a few more isolated outliers, namely observations 12, 13, 31 and 32.

`bushfirem` is created from bushfire by setting a proportion of 0.2 of the values to missing.

For testing purposes weights are provided: `bushfire.weights <- rep(c(1,2,5), length = nrow(bushfire))`

References

Campbell, N. (1989) Bushfire Mapping using NOAA AVHRR Data. Technical Report. Commonwealth Scientific and Industrial Research Organisation, North Ryde.

Examples

```
data(bushfirem)
```

EAdet *Epidemic Algorithm for detection of multivariate outliers in incomplete survey data*

Description

In `EAdet` an epidemic is started at a center of the data. The epidemic spreads out and infects neighbouring points (probabilistically or deterministically). The last points infected are outliers. After running `EAdet` an imputation with `EAimp` may be run.

Usage

```
EAdet(
  data,
  weights,
  reach = "max",
  transmission.function = "root",
  power = ncol(data),
  distance.type = "euclidean",
  maxl = 5,
  plotting = TRUE,
  monitor = FALSE,
  prob.quantile = 0.9,
  random.start = FALSE,
  fix.start,
  threshold = FALSE,
  deterministic = TRUE,
  rm.missobs = FALSE,
  verbose = FALSE
)
```

Arguments

`data` a data frame or matrix with data.

`weights` a vector of positive sampling weights.

`reach` if `reach = "max"` the maximal nearest neighbor distance is used as the basis for the transmission function, otherwise the weighted $(1-(p+1)/n)$ quantile of the nearest neighbor distances is used.

`transmission.function` form of the transmission function of distance d: `"step"` is a heaviside function which jumps to `1` at `d0`, `"linear"` is linear between `0` and `d0`, `"power"` is `(beta*d+1)^(-p)` for `p = ncol(data)` and `beta <- as.single((0.01^(-1 / power) - 1) / d0))` as default, `"root"` is the function `1-(1-d/d0)^(1/maxl)`.

`power` sets `p = power`.

`distance.type` distance type in function `dist()`.

`maxl` maximum number of steps without infection.

`plotting` if `TRUE`, the cdf of infection times is plotted.

`monitor` if `TRUE`, verbose output on epidemic.

`prob.quantile` if mads fail, take this quantile absolute deviation.

`random.start` if `TRUE`, take a starting point at random instead of the spatial median.

`fix.start` force epidemic to start at a specific observation.

`threshold` infect all remaining points with infection probability above the threshold `1-0.5^(1/maxl)`.

`deterministic` if `TRUE`, the number of infections is the expected number and the infected observations are the ones with largest infection probabilities.

`rm.missobs` set `rm.missobs=TRUE` if completely missing observations should be discarded. This has to be done actively as a safeguard to avoid mismatches when imputing.

`verbose` more output with `verbose=TRUE`.

Details

The form and parameters of the transmission function should be chosen such that the infection times have at least a range of 10. The default cutting point to decide on outliers is the median infection time plus three times the mad of infection times. A better cutpoint may be chosen by visual inspection of the cdf of infection times. `EAdet` calls the function `EA.dist`, which passes the counterprobabilities of infection (a $n * (n-1)/2$ size vector!) and three parameters (sample spatial median index, maximal distance to nearest neighbor and transmission distance = reach) as arguments to `EAdet`. The distances vector may be too large to be passed as arguments. Then either the memory size must be increased. Former versions of the code used a global variable to store the distances in order to save memory.

Value

`EAdet` returns a list whose first component `output` is a sub-list with the following components:

`sample.size` Number of observations

`discarded.observations` Indices of discarded observations

`missing.observations` Indices of completely missing observations

`number.of.variables` Number of variables

`n.complete.records` Number of records without missing values

`n.usable.records` Number of records with less than half of values missing (unusable observations are discarded)

`medians` Component wise medians

`mads` Component wise mads

`prob.quantile` Use this quantile if mads fail, i.e. if one of the mads is 0

`quantile.deviations` Quantile of absolute deviations

`start` Starting observation

`transmission.function` Input parameter

`power` Input parameter

`maxl` Maximum number of steps without infection

`min.nn.dist` Maximal nearest neighbor distance

`transmission.distance` `d0`

`threshold` Input parameter

`distance.type` Input parameter

`deterministic` Input parameter

`number.infected` Number of infected observations

`cutpoint` Cutpoint of infection times for outlier definition

`number.outliers` Number of outliers

`outliers` Indices of outliers

`duration` Duration of epidemic

`computation.time` Elapsed computation time `initialisation.computation.time` Elapsed computation time for standardisation and calculation of distance matrix

The further components returned by `EAdet` are:

`infected` Indicator of infection

`infection.time` Time of infection

`outind` Indicator of outliers

Author(s)

Beat Hulliger

References

Béguin, C. and Hulliger, B. (2004) Multivariate outlier detection in incomplete survey data: the epidemic algorithm and transformed rank correlations, JRSS-A, 167, Part 2, pp. 275-294.

See Also

`EAimp` for imputation with the Epidemic Algorithm.

Examples

```
data(bushfirem, bushfire.weights)
det.res <- EAdet(bushfirem, bushfire.weights)
```

`EAimp` *Epidemic Algorithm for imputation of multivariate outliers in incomplete survey data.*

Description

After running `EAdet` an imputation of the detected outliers with `EAimp` may be run.

Usage

```
EAimp(
  data,
  weights,
  outind,
  reach = "max",
  transmission.function = "root",
  power = ncol(data),
  distance.type = "euclidean",
  duration = 5,
  maxl = 5,
  kdon = 1,
  monitor = FALSE,
  threshold = FALSE,
  deterministic = TRUE,
  fixedprop = 0
)
```

Arguments

`data` a data frame or matrix with the data.

`weights` a vector of positive sampling weights.

`outind` a logical vector with component `TRUE` for outliers.

`reach` reach of the threshold function (usually set to the maximum distance to a nearest neighbour, see internal function `EA.dist`).

`transmission.function` form of the transmission function of distance d: `"step"` is a heaviside function which jumps to `1` at `d0`, `"linear"` is linear between `0` and `d0`, `"power"` is `beta*d+1^(-p)` for `p=ncol(data)` as default, `"root"` is the function `1-(1-d/d0)^(1/maxl)`.

`power` sets `p=power`, where `p` is the parameter in the above transmission function.

`distance.type` distance type in function `dist()`.

`duration` the duration of the detection epidemic.

`maxl` maximum number of steps without infection.

`kdon` the number of donors that should be infected before imputation.

`monitor` if `TRUE` verbose output on epidemic.

`threshold` Infect all remaining points with infection probability above the threshold `1-0.5^(1/maxl)`.

`deterministic` if `TRUE` the number of infections is the expected number and the infected observations are the ones with largest infection probabilities.

`fixedprop` if `TRUE` a fixed proportion of observations is infected at each step.

Details

`EAimp` uses the distances calculated in `EAdet` (actually the counterprobabilities, which are stored in a global data set) and starts an epidemic at each observation to be imputed until donors for the missing values are infected. Then a donor is selected randomly.

Value

`EAimp` returns a list with two components: `parameters` and `imputed.data`. `parameters` contains the following elements:

`sample.size` Number of observations

`number.of.variables` Number of variables

`n.complete.records` Number of records without missing values

`n.usable.records` Number of records with less than half of values missing (unusable observations are discarded) `duration` Duration of epidemic

`reach` Transmission distance (`d0`)

`threshold` Input parameter

`deterministic` Input parameter

`computation.time` Elapsed computation time

`imputed.data` contains the imputed data.

Author(s)

Beat Hulliger

References

Béguin, C. and Hulliger, B. (2004) Multivariate outlier detection in incomplete survey data: the epidemic algorithm and transformed rank correlations, JRSS-A, 167, Part 2, pp. 275-294.

See Also

`EAdet` for outlier detection with the Epidemic Algorithm.

Examples

```
data(bushfirem, bushfire.weights)
det.res <- EAdet(bushfirem, bushfire.weights)
imp.res <- EAimp(bushfirem, bushfire.weights, outind = det.res$outind, kdon = 3)
print(imp.res$output)
```

ER *Robust EM-algorithm ER*

Description

The `ER` function is an implementation of the ER-algorithm of Little and Smith (1987).

Usage

```
ER(
  data,
  weights,
  alpha = 0.01,
  psi.par = c(2, 1.25),
  em.steps = 100,
  steps.output = FALSE,
  Estep.output = FALSE,
  tolerance = 1e-06
)
```

Arguments

`data`	a data frame or matrix with the data.
`weights`	sampling weights.
`alpha`	probability for the quantile of the cut-off.
`psi.par`	further parameters passed to the psi-function.
`em.steps`	number of iteration steps of the EM-algorithm.
`steps.output`	if `TRUE`, verbose output.
`Estep.output`	if `TRUE`, estimators are output at each iteration.
`tolerance`	convergence criterion (relative change).

Details

The M-step of the EM-algorithm uses a one-step M-estimator.

Value

`sample.size` Number of observations

`number.of.variables` Number of variables

`significance.level` alpha

`computation.time` Elapsed computation time

`good.data` Indices of the data in the final good subset

`outliers` Indices of the outliers

`center` Final estimate of the center

`scatter` Final estimate of the covariance matrix

`dist` Final Mahalanobis distances

`rob.weights` Robustness weights in the final EM step

Author(s)

Beat Hulliger

References

Little, R. and P. Smith (1987). Editing and imputation for quantitative survey data. Journal of the American Statistical Association, 82, 58-68.

See Also

`BEM`

Examples

```
data(bushfirem, bushfire.weights)
det.res <- ER(bushfirem, weights = bushfire.weights, alpha = 0.05,
steps.output = TRUE, em.steps = 100, tol = 2e-6)
PlotMD(det.res$dist, ncol(bushfirem))
```

GIMCD *Gaussian imputation followed by MCD*

Description

Gaussian imputation uses the classical non-robust mean and covariance estimator and then imputes predictions under the multivariate normal model. Outliers may be created by this procedure. Then a high-breakdown robust estimate of the location and scatter with the Minimum Covariance Determinant algorithm is obtained and finally outliers are determined based on Mahalanobis distances based on the robust location and scatter.

Usage

```
GIMCD(data, alpha = 0.05, seedem = 23456789, seedmcd)
```

Arguments

`data`	a data frame or matrix with the data.
`alpha`	a threshold value for the cut-off for the outlier Mahalanobis distances.
`seedem`	random number generator seed for EM algorithm
`seedmcd`	random number generator seed for MCD algorithm, if `seedmcd` is missing, an internal seed will be used.

Details

Normal imputation from package `norm` and MCD from package `MASS`. Note that currently MCD does not accept weights.

Value

Result is stored in a global list GIMCD.r:

`center` robust center

`scatter` robust covariance

`alpha` quantile for cut-off value

`computation.time` elapsed computation time

`outind` logical vector of outlier indicators

`dist` Mahalanobis distances

Author(s)

Beat Hulliger

References

Béguin, C. and Hulliger, B. (2008), The BACON-EEM Algorithm for Multivariate Outlier Detection, in Incomplete Survey Data, Survey Methodology, Vol. 34, No. 1, pp. 91-103.

See Also

`cov.rob`

Examples

```
data(bushfirem)
det.res <- GIMCD(bushfirem, alpha = 0.1)
print(det.res$center)
PlotMD(det.res$dist, ncol(bushfirem))
```

`lival` *Living Standards Measurement Survey Albania 2012*

Description

The dataset is an extended version of the public micro data file of the LSMS 2012 of Albania available at (`https://www.instat.gov.al/en/figures/micro-data/`, accessed 13 February 2023). Documentation of the LSMS 2012 of Albania is from the World Bank (`https://microdata.worldbank.org/index.php/catalog/1970`, accessed 5 November 2020). The data set is ported to R and updated with approximate survey design information derived from the data itself. The units are households and the variables are expenditures on main categories, poverty measures and structural information including weights and sample design.

Usage

```
lival
```

Format

A data frame with 6671 rows and 26 variables

psu primary sampling unit (psu)

hhid unique household identifier (100*psu+hh)

hh household number per psu

prefectu prefecture

urban urbanicity (Urban=1, Rural=2)

strat stratum

region region

totcons total consumption of hh

rcons real mean per capita consumption

rfood real food consumption per capita

rtotnfoo real non food consumption per capita

reduexpp real education consumption per capita

rdurcons real durable consumption per capita

rtotutil real utilities consumption per capita

egap0 extreme headcount poverty

egap1 extreme poverty gap

egap2 extreme poverty depth

agap0 absolute headcount poverty

agap1 absolute poverty gap

agap2 absolute poverty depth

weight final cross-sectional weight

nph number of psu in stratum population

mph number of households in stratum population

mphi number of households in sampled psu

pi1 psu inclusion probability

pi2 household inclusion probability

Details

Absolute poverty measures use a poverty line of Lek 4891 (2002 prices). Extreme poverty measures use a poverty line where the basic nutritional needs are difficult to meet. The headcount poverty variable is an indicator for the income of the household y_i being below the (absolute or extreme) poverty line z. The poverty gap variable measures the relative distance to the poverty line: $(z - y_i)/z$. The poverty depth variable is the square of the poverty gap variable, i.e. $[(z-y_i)/z]^2$, giving more weight to the poorer among the poor and thus describing the inequality among the poor.

The survey design is a stratified clustered two stage design. The primary sampling units are enumeration zones. The strata are the crossing of prefecture and urbanicity and the allocation of the psu sample to the strata is proportional to the number of households. Within strata the psu are sampled with probability proportional to number of households. Within psu a simple random sample of 8 households was selected. The weights are calibrated to population margins. All survey design informations except the strata and the weights are approximated through the weights using assumptions on the design. Since the data set has undergone data protection measures and the survey design is approximate only, inference to the population does not yield exact results. However, the complexity of the data and of the survey design are realistic.

The size of the household is not on the original data set. However, the transformation `capita <- round(0.07527689 * totcons/rcons, 0)` yields the number of persons in the household.

Note

With R package survey a survey design object can be built with, e.g., `svydesign(~psu + hhid , strata= ~strat, fpc= ~pi1 +pi2, weight= ~weight, data=lival, pps="brewer")`.

References

https://www.instat.gov.al/en/figures/micro-data/

Examples

```
data(lival)
lival$capita <- with(lival, round(0.07527689 * totcons / rcons, 0))
## Not run:
library(survey)
lival.des <- svydesign(~psu + hhid , strata= ~strat, fpc= ~pi1 +pi2,
                       weight= ~weight, data=lival, pps="brewer")
svymean(~totcons, lival.des, deff=TRUE)

## End(Not run)
```

MDmiss *Mahalanobis distance (MD) for data with missing values*

Description

For each observation the missing dimensions are omitted before calculating the MD. The MD contains a correction factor p/q to account for the number of observed values, where p is the number of variables and q is the number of observed dimensions for the particular observation.

Usage

```
MDmiss(data, center, cov)
```

Arguments

data	the data as a dataframe or matrix.
center	the center to be used (may not contain missing values).
cov	the covariance to be used (may not contain missing values).

Details

The function loops over the observations. This is not optimal if only a few missingness patterns occur. If no missing values occur the function returns the Mahalanobis distance.

Value

The function returns a vector of the (squared) Mahalanobis distances.

Author(s)

Beat Hulliger

References

Béguin, C., and Hulliger, B. (2004). Multivariate outlier detection in incomplete survey data: The epidemic algorithm and transformed rank correlations. Journal of the Royal Statistical Society, A167 (Part 2.), pp. 275-294.

See Also

mahalanobis

Examples

```
data(bushfirem, bushfire)
MDmiss(bushfirem, apply(bushfire, 2, mean), var(bushfire))
```

modi *modi: Multivariate outlier detection for incomplete survey data.*

Description

The package modi is a collection of functions for multivariate outlier detection and imputation. The aim is to provide a set of functions which cope with missing values and take sampling weights into account. The original functions were developed in the EUREDIT project. This work was partially supported by the EU FP5 ICT programme, the Swiss Federal Office of Education and Science and the Swiss Federal Statistical Office. Subsequent development was in the AMELI project of the EU FP7 SSH Programme and also supported by the University of Applied Sciences and Arts Northwestern Switzerland (FHNW).

modi functions

BACON-EEM algorithm in `BEM()`, Epidemic algorithm in `EAdet()` and `EAimp()`, Transformed Rank Correlations in `TRC()`, Gaussian imputation with MCD in `GIMCD()`.

References

Béguin, C., and Hulliger, B. (2004). Multivariate outlier detection in incomplete survey data: The epidemic algorithm and transformed rank correlations. Journal of the Royal Statistical Society, A167 (Part 2.), pp. 275-294.

Béguin, C., and Hulliger, B. (2008). The BACON-EEM Algorithm for Multivariate Outlier Detection in Incomplete Survey Data, Survey Methodology, Vol. 34, No. 1, pp. 91-103.

plotIT *Plot of infection times of the EA algorithm*

Description

The (weighted) cdf of infection times is plotted. The infection times jumps of the cdf are shown by the points with the same infection times stacked vertically and respecting the weights.

Usage

```
plotIT(infection.time, weights, cutpoint)
```

Arguments

`infection.time`	vector of infection.times of the observations
`weights`	vector of (survey) weights of the observations
`cutpoint`	a cutpoint to for declaring outliers

Details

The infection times of `EAdet` are the main input. In addition the weights may be needed. The default cutpoint from `EAdet` may be used for the cutpoint. Points that are never infected have a missing infection time. These missing infection times are (temporarily) imputed by 1.2 times the maximum infection time to show them on the plot marked with an x.

Author(s)

Beat Hulliger

Examples

```
it <- c(rep(NA, 3), rep(1:7, times=c(1, 4, 10, 8, 5, 3, 2)))
wt <- rep(c(1,2,5), times=12)
plotIT(it, wt, 6)
```

`PlotMD` *QQ-Plot of Mahalanobis distances*

Description

QQ-plot of (squared) Mahalanobis distances vs. scaled F-distribution (or a scaled chisquare distribution). In addition, two default cutpoints are proposed.

Usage

```
PlotMD(dist, p, alpha = 0.95, chisquare = FALSE)
```

Arguments

`dist`	a vector of Mahalanobis distances.
`p`	the number of variables involved in the Mahalanobis distances.
`alpha`	a probability for cut-off, usually close to 1.
`chisquare`	a logical indicating the the chisquare distribution should be used instead of the F-distribution.

Details

Scaling of the F-distribution as `median(dist)*qf((1:n)/(n+1), p, n-p)/qf(0.5, p, n-p)`. First default cutpoint is `median(dist)*qf(alpha, p, n-p)/qf(0.5, p, n-p)` and the second default cutpoint is the alpha quantile of the Mahalanobis distances.

Value

`hmed` first proposed cutpoint based on F-distribution

`halpha` second proposed cutpoint (alpha-quantile)

`QQ-plot`

Author(s)

Beat Hulliger

References

Little, R. & Smith, P. (1987) Editing and imputation for quantitative survey data, Journal of the American Statistical Association, 82, 58-68

Examples

```
data(bushfirem, bushfire.weights)
det.res <- TRC(bushfirem, weights = bushfire.weights)
PlotMD(det.res$dist, ncol(bushfirem))
```

`POEM` *Nearest Neighbour Imputation with Mahalanobis distance*

Description

POEM takes into account missing values, outlier indicators, error indicators and sampling weights.

Usage

```
POEM(
  data,
  weights,
  outind,
  errors,
  missing.matrix,
  alpha = 0.5,
  beta = 0.5,
  reweight.out = FALSE,
  c = 5,
  preliminary.mean.imputation = FALSE,
  monitor = FALSE
)
```

Arguments

`data` a data frame or matrix with the data.

`weights` sampling weights.

`outind` an indicator vector for the outliers with `1` indicating an outlier.

`errors` matrix of indicators for items which failed edits.

`missing.matrix` the missingness matrix can be given as input. Otherwise, it will be recalculated.

`alpha` scalar giving the weight attributed to an item that is failing.

`beta` minimal overlap to accept a donor.

`reweight.out` if `TRUE`, the outliers are redefined.

`c` tuning constant when redefining the outliers (cutoff for Mahalanobis distance).

`preliminary.mean.imputation`
assume the problematic observation is at the mean of good observations.

`monitor` if `TRUE` verbose output.

Details

`POEM` assumes that an multivariate outlier detection has been carried out beforehand and assumes the result is summarized in the vector `outind`. In addition, further observations may have been flagged as failing edit-rules and this information is given in the vector `errors`. The mean and covariance estimate is calculated with the good observations (no outliers and downweighted errors). Preliminary mean imputation is sometimes needed to avoid a non-positive definite covariance estimate at this stage. Preliminary mean imputation assumes that the problematic values of an observation (with errors, outliers or missing) can be replaced by the mean of the rest of the non-problematic observations. Note that the algorithm imputes these problematic observations afterwards and therefore the final covariance matrix with imputed data is not the same as the working covariance matrix (which may be based on preliminary mean imputation).

Value

`POEM` returns a list whose first component `output` is a sub-list with the following components:

`preliminary.mean.imputation` Logical. `TRUE` if preliminary mean imputation should be used

`completely.missing` Number of observations with no observed values

`good.values` Weighted number of of good values (not missing, not outlying, not erroneous)

`nonoutliers.before` Number of nonoutliers before reweighting

`weighted.nonoutliers.before` Weighted number of nonoutliers before reweighting

`nonoutliers.after` Number of nonoutliers after reweighting

`weighted.nonoutliers.after` Weighted number of nonoutliers after reweighting

`old.center` Coordinate means after weighting, before imputation

`old.variances` Coordinate variances after weighting, before imputation

`new.center` Coordinate means after weighting, after imputation

`new.variances` Coordinate variances after weighting, after imputation `covariance` Covariance (of standardised observations) before imputation

`imputed.observations` Indices of observations with imputed values

`donors` Indices of donors for imputed observations

`new.outind` Indices of new outliers

The further component returned by `POEM` is:

`imputed.data` Imputed data set

Author(s)

Beat Hulliger

References

Béguin, C. and Hulliger B., (2002), EUREDIT Workpackage x.2 D4-5.2.1-2.C Develop and evaluate new methods for statistical outlier detection and outlier robust multivariate imputation, Technical report, EUREDIT 2002.

Examples

```
data(bushfirem, bushfire.weights)
outliers <- rep(0, nrow(bushfirem))
outliers[31:38] <- 1
imp.res <- POEM(bushfirem, bushfire.weights, outliers,
preliminary.mean.imputation = TRUE)
print(imp.res$output)
var(imp.res$imputed.data)
```

`sepe` *Sample Environment Protection Expenditure Survey.*

Description

The sepe data set is a sample of the pilot survey in 1993 of the Swiss Federal Statistical Office on environment protection expenditures of Swiss private economy in the previous accounting year. The units are enterprises, the monetary variables are in thousand Swiss Francs (CHF). From the original sample a random subsample was chosen of which certain enterprises were excluded for confidentiality reasons. In addition, noise has been added to certain variables, and certain categories have been collapsed. The data set has missing values. The data set has first been prepared for the EU FP5 project EUREDIT and later been data protected for educational purposes.

Usage

```
sepe
```

Format

A data frame with 675 rows and 23 variables:

idnr identifier (anonymous)

exp categorical variable where 1 = 'non-zero total expenditure' and 2 = 'zero total expenditure, and 3 = 'no answer'

totinvwp total investment for water protection

totinvwm total investment for waste management

totinvap total investment for air protection

totinvnp total investment for noise protection

totinvot total investment for other environmental protection

totinvto overall total investment in all environmental protection areas

totexpwp total current expenditure in environmental protection area water protection

totexpwm total current expenditure in environmental protection area waste management

totexpap total current expenditure in environmental protection area air protection

totexpnp total current expenditure in environmental protection area noise protection

totexpot total current expenditure in other environmental protection

totexpto overall total current expenditure in all environmental protection

subtot total subsidies for environmental protection received

rectot total receipts from environmental protection

employ number of employees

sizeclass size class (according to number of employees)

stratum stratum number of sample design

activity code of economic activity (aggregated)

popsize number of enterprises in the population-stratum

popempl number of employees in population activity group

weight sampling weight (for extrapolation to the population)

Details

The sample design is stratified random sampling with different sampling rates. Use package survey or sampling to obtain correct point and variance estimates. In addition a ratio estimator may be built using the variable popemple which gives the total employment per activity.

There are two balance rules: the subtotals of the investment variables should sum to totinvto and the expenditure subtotals should sum to totexpto.

The missing values stem from the survey itself. In the actual survey the missing values were declared as 'guessed' rather than copied from records.

The sampling weight weight is adjusted for non-response in the stratum, i.e. `weight=popsize/sampsize`.

References

Swiss Federal Statistical Office (1996), Umweltausgaben und -investitionen in der Schweiz 1992/1993, Ergebnisse einer Pilotstudie.

Charlton, J. (ed.), Towards Effective Statistical Editing and Imputation Strategies - Findings of the Euredit project, unpublished manuscript available from Eurostat and `https://www.cs.york.ac.uk/euredit/euredit-main.html`.

Examples

```
data(sepe)
```

`TRC` *Transformed rank correlations for multivariate outlier detection*

Description

`TRC` starts from bivariate Spearman correlations and obtains a positive definite covariance matrix by back-transforming robust univariate medians and mads of the eigenspace. `TRC` can cope with missing values by a regression imputation using the a robust regression on the best predictor and it takes sampling weights into account.

Usage

```
TRC(
  data,
  weights,
  overlap = 3,
  mincor = 0,
  robust.regression = "rank",
  gamma = 0.5,
  prob.quantile = 0.75,
  alpha = 0.05,
  md.type = "m",
  monitor = FALSE
)
```

Arguments

`data`	a data frame or matrix with the data.
`weights`	sampling weights.
`overlap`	minimum number of jointly observed values for calculating the rank correlation.
`mincor`	minimal absolute correlation to impute.
`robust.regression`	type of regression: `"irls"` is iteratively reweighted least squares M-estimator, `"rank"` is based on the rank correlations.

`gamma`	minimal number of jointly observed values to impute.
`prob.quantile`	if mads are 0, try this quantile of absolute deviations.
`alpha`	`(1 - alpha)` Quantile of F-distribution is used for cut-off.
`md.type`	type of Mahalanobis distance when missing values occur: `"m"` marginal (default), `"c"` conditional.
`monitor`	if `TRUE`, verbose output.

Details

`TRC` is similar to a one-step OGK estimator where the starting covariances are obtained from rank correlations and an ad hoc missing value imputation plus weighting is provided.

Value

`TRC` returns a list whose first component `output` is a sublist with the following components:

`sample.size` Number of observations

`number.of.variables` Number of variables

`number.of.missing.items` Number of missing values

`significance.level` `1 - alpha`

`computation.time` Elapsed computation time

`medians` Componentwise medians

`mads` Componentwise mads

`center` Location estimate

`scatter` Covariance estimate

`robust.regression` Input parameter

`md.type` Input parameter

`cutpoint` The default threshold MD-value for the cut-off of outliers

The further components returned by `TRC` are:

`outind` Indicator of outliers

`dist` Mahalanobis distances (with missing values)

Author(s)

Beat Hulliger

References

Béguin, C. and Hulliger, B. (2004) Multivariate outlier detection in incomplete survey data: the epidemic algorithm and transformed rank correlations, JRSS-A, 167, Part 2, pp. 275-294.

Examples

```
data(bushfirem, bushfire.weights)
det.res <- TRC(bushfirem, weights = bushfire.weights)
PlotMD(det.res$dist, ncol(bushfirem))
print(det.res)
```

`weighted.quantile` *Quantiles of a weighted cdf*

Description

A weighted cdf is calculated and quantiles are evaluated. Missing values are discarded.

Usage

```
weighted.quantile(x, w, prob = 0.5, plot = FALSE)
```

Arguments

`x`	a vector of data.
`w`	a vector of (sampling) weights.
`prob`	the probability for the quantile.
`plot`	if `TRUE`, the weighted cdf is plotted.

Details

Weighted linear interpolation in case of non-unique inverse. Gives a warning when the contribution of the weight of the smallest observation to the total weight is larger than `prob`.

Value

The quantile according to `prob` (by default it returns the weighted median).

Note

No variance calculation.

Author(s)

Beat Hulliger

See Also

svyquantile

Examples

```
x <- rnorm(100)
x[sample(1:100, 20)] <- NA
w <- rchisq(100, 2)
weighted.quantile(x, w, 0.2, TRUE)
```

---

weighted.var *Weighted univariate variance coping with missing values*

Description

This function is analogous to weighted.mean.

Usage

```
weighted.var(x, w, na.rm = FALSE)
```

Arguments

x	a vector of data.
w	a vector of positive weights (may not have missings where x is observed).
na.rm	if TRUE remove missing values.

Details

The weights are standardised such that $\sum_{observed} w_i$ equals the number of observed values in x. The function calculates

$$\sum_{observed} w_i(x_i - weighted.mean(x, w, na.rm = TRUE))^2/((\sum_{observed} w_i) - 1)$$

Value

The weighted variance of x with weights w (with missing values removed when na.rm = TRUE).

Author(s)

Beat Hulliger

See Also

weighted.mean

Examples

```
x <- rnorm(100)
x[sample(1:100, 20)] <- NA
w <- rchisq(100, 2)
weighted.var(x, w, na.rm = TRUE)
```

`Winsimp` *Winsorization followed by imputation*

Description

Winsorization of outliers according to the Mahalanobis distance followed by an imputation under the multivariate normal model. Only the outliers are winsorized. The Mahalanobis distance MD-miss allows for missing values.

Usage

```
Winsimp(data, center, scatter, outind, seed = 1000003)
```

Arguments

`data`	a data frame with the data.
`center`	(robust) estimate of the center (location) of the observations.
`scatter`	(robust) estimate of the scatter (covariance-matrix) of the observations.
`outind`	logical vector indicating outliers with 1 or TRUE for outliers.
`seed`	seed for random number generator.

Details

It is assumed that `center`, `scatter` and `outind` stem from a multivariate outlier detection algorithm which produces robust estimates and which declares outliers observations with a large Mahalanobis distance. The cutpoint is calculated as the least (unsquared) Mahalanobis distance among the outliers. The winsorization reduces the weight of the outliers:

$$\hat{y}_i = \mu_R + (y_i - \mu_R) \cdot c/d_i$$

where μ_R is the robust center and d_i is the (unsquared) Mahalanobis distance of observation i.

Value

`Winsimp` returns a list whose first component `output` is a sublist with the following components:

`cutpoint` Cutpoint for outliers

`proc.time` Processing time

`n.missing.before` Number of missing values before imputation

`n.missing.after` Number of missing values after imputation

The further component returned by `winsimp` is:

`imputed.data` Imputed data set

Author(s)

Beat Hulliger

References

Hulliger, B. (2007), Multivariate Outlier Detection and Treatment in Business Surveys, Proceedings of the III International Conference on Establishment Surveys, Montréal.

See Also

MDmiss. Uses imp.norm.

Examples

```
data(bushfirem, bushfire.weights)
det.res <- TRC(bushfirem, weight = bushfire.weights)
imp.res <- Winsimp(bushfirem, det.res$center, det.res$scatter, det.res$outind)
print(imp.res$n.missing.after)
```

Index