cem: Coarsened Exact Matching

Version 1.0.1
August 5, 2008

Contents

1 Introduction

This program is designed to improve the estimation of causal effects via a powerful method of matching that is widely applicable in observational data and exceptionally easy to understand and use (if you understand how to draw a histogram, you will understand this method). The program implements the CEM (Coarsened Exact Matching) algorithm described in

Stefano M. Iacus, Gary King, and Giuseppe Porro, "Matching for Causal Inference Without Balance Checking", copy at `http://gking.harvard.edu/files/abs/cem-abs.shtml`.

CEM is a monotonoic imbalance reducing matching method — which means that the balance between the treated and control groups is chosen by the user ex ante rather than discovered through the usual laborious process of checking after the fact and repeatdly reestimating, and so that adjusting the imbalance on one variable has no effect on the maximum imbalance of any other. CEM also strictly bounds through ex ante user choice both the degree of model dependence and the average treatment effect estimation error, eliminates the need for a separate procedure to restrict data to common empirical support, meets the congruence principle, is robust to measurement error, works well with multiple imputation methods for missing data, can be completely automated, and is extremely fast computationally even with very large data sets. After preprocessing data with CEM, the analyst may then use a simple difference in means or whatever statistical model they would have applied without matching. CEM also works well for multicategory treatments, determining blocks in experimental designs, and evaluating extreme counterfactuals.

2 Software Requirements

CEM works in conjunction with the R Project for Statistical Computing, and will run on any platform where R is installed (Windows, Unix, or Mac). R is available free for download at the Comprehensive R Archive Network (CRAN) at `http://cran.r-project.org/`. CEM has been tested on the most recent version of R.

CEM may be run by installing the program directly, as indicated below, or by using the alternative interface to CEM provided by MatchIt (`http://gking.harvard.edu/matchit`, (Ho et al., Forthcoming)). Using CEM directly is faster. The MatchIt interface is easier for some applications and also works seemlessly with Zelig (`http://gking.harvard.edu/zelig`) for estimating causal effects after matching. (A Stata verison of CEM is under development.)

3 Installation

To install cem, type at the R command prompt,

```
> install.packages("cem")
```

and CEM will install itself onto your system automatically from CRAN. (You may alternatively load the beta test version as

```
> install.packages("cem",repos="http://gking.harvard.edu/cem")
```

4 Loading CEM

You need to install CEM only once, but you must load it prior to each use. Do this at the R prompt:

```
> library(cem)
```

5 Updating CEM

We recommend that you periodically update CEM at the R prompt by typing:

```
> update.packages()
> library(cem)
```

which will update all the libraries including CEM and load the new version of CEM.

6 R Functions

6.1 cem: Coarsened Exact Matching

Description

Implementation of Coarsened Exact Matching

Usage

```
cem(treatment = NULL, data, cutpoints = NULL, drop = NULL,
    eval.imbalance = FALSE, k2k = FALSE,  method=NULL, mpower=2, verbose = 0)
```

Arguments

`treatment`	character, name of the treatment variable
`data`	a data.frame
`cutpoints`	named list each describing the cutpoints for the variables (the names are variable names). Each list element is either a vector of cutpoints, a number of cutpoints, or a method for automatic bin contruction. See Details.
`drop`	a vector of variable names in the data frame to ignore during matching
`eval.imbalance`	Boolean. See Details.
`k2k`	boolean, return k-to-k matching? Default = `FALSE`
`method`	distance method to use in `k2k` matching. See Details.
`mpower`	power of the Minkowski distance. See Details.
`verbose`	controls level of verbosity. Default=0.

Details

When specifying cutpoints, several automatic methods can be chosen among "`sturges`" (Surges' rule, the default), "`fd`" (Freedman-Diaconis' rule), "`scott`" (Scott's rule) and "ss" (Shimazaki-Shinomoto's rule). See references for a description of each rule.

`verbose`: a number greater or equal to 0. The higher, the more info are provided during the execution of the algorithm.

If `eval.imbalance` = `TRUE`, `cem$imbalance` contains the imbalance measure by absolute difference in means for numerical variables and chi-square distance for categorical variables. If `FALSE` (the default) then `cem$imbalance` is set to `NULL`.

If `k2k` is set to `TRUE`, the algorithm return strata with the same number of treated and control units per stratum, otherwise all the matched units are returned (default). When `k2k` = `TRUE`, the user can choose a `method` (between '`euclidean`', '`maximum`', '`manhattan`', '`canberra`', '`binary`' and '`minkowski`') for nearest neighbor matching inside each `cem` strata. By default `method` is set to '`NULL`', which means random matching inside `cem`

strata. For the Minkowski distance the power can be specified via the argument `mpower`'. For more information on `method != NULL`, refer to `dist` help page.

In case of missing data, `cem` gives a warning and treats missing values as distinct values and match observations with missing values in the same variable in the same stratum provided that all the remaining (corasened) covariates match.

Value

`call`	the call
`strata`	vector of stratum number in which each observation belongs, NA if the observation has not been matched
`n.strata`	number of strata generated
`vars`	report variables names used for the match
`drop`	variables removed from the match
`breaks`	named list of cutpoints, eventually NULL
`treatment`	name of the treatment variable
`groups`	factor, each observation belong to one group generated by the treatment variable
`n.groups`	number of groups identified by the treatment variable
`group.idx`	named list, index of observations belonging to each group
`group.len`	sizes of groups
`tab`	summary table of matched by group
`imbalance`	NULL or a vector of imbalances. See Details.

Author(s)

Stefano Iacus, Gary King, and Giusseppe Porro

References

Stefano Iacus, Gary King, Giusseppe Porro, "Matching for Casual Inference Without Balance Checking," http://gking.harvard.edu/files/abs/cem-abs.shtml

Examples

```
data(LL)

mybr = list(re74=hist(LL$re74,plot=FALSE)$breaks,
 re75 = hist(LL$re75,plot=FALSE)$breaks,
 age = hist(LL$age,plot=FALSE)$breaks,
```

```
 education = hist(LL$education,plot=FALSE)$breaks)

L1.meas(LL$treated, LL[,-c(1,9)], breaks=mybr)
eval.match(LL$treated, LL[,-c(1,9)], breaks=mybr)

# cem match: automatic bin choice
mat <- cem(treatment="treated",data=LL, drop="re78")
mat$tab
cem1.idx <- which(mat$matched)
# imbalance
L1.meas(LL$treated[cem1.idx], LL[cem1.idx,-c(1,9)], breaks=mybr)
eval.match(LL$treated[cem1.idx], LL[cem1.idx,-c(1,9)], breaks=mybr)
L1.meas(LL$treated[cem1.idx], LL[cem1.idx,-c(1,9)], breaks=mybr,weights=mat$w[cem1.idx])
eval.match(LL$treated[cem1.idx], LL[cem1.idx,-c(1,9)], breaks=mybr,weights=mat$w[cem1.idx])

# cem match: user choiced coarsening
re74cut <- hist(LL$re74, br=seq(0,max(LL$re74)+1000, by=1000),plot=FALSE)$breaks
re75cut <- hist(LL$re75, br=seq(0,max(LL$re75)+1000, by=1000),plot=FALSE)$breaks
agecut <- hist(LL$age, br=seq(15,55, length=14),plot=FALSE)$breaks
mycp <- list(re75=re75cut, re74=re74cut, age=agecut)
mat <- cem(treatment="treated",data=LL, drop="re78",cutpoints=mycp)
mat$tab
cem2.idx <- which(mat$matched)
#imbalance
L1.meas(LL$treated[cem2.idx], LL[cem2.idx,-c(1,9)], breaks=mybr)
eval.match(LL$treated[cem2.idx], LL[cem2.idx,-c(1,9)], breaks=mybr)
L1.meas(LL$treated[cem2.idx], LL[cem2.idx,-c(1,9)], breaks=mybr,weights=mat$w[cem2.idx])
eval.match(LL$treated[cem2.idx], LL[cem2.idx,-c(1,9)], breaks=mybr,weights=mat$w[cem2.idx])

# cem match: user choiced coarsening, k-to-k matching
mat <- cem(treatment="treated",data=LL, drop="re78",cutpoints=mycp,k2k=TRUE)
mat$tab
cem3.idx <- which(mat$matched)
#imbalance
L1.meas(LL$treated[cem3.idx], LL[cem3.idx,-c(1,9)], breaks=mybr)
eval.match(LL$treated[cem3.idx], LL[cem3.idx,-c(1,9)], breaks=mybr)
L1.meas(LL$treated[cem3.idx], LL[cem3.idx,-c(1,9)], breaks=mybr,weights=mat$w[cem3.idx])
eval.match(LL$treated[cem3.idx], LL[cem3.idx,-c(1,9)], breaks=mybr,weights=mat$w[cem3.idx])

# mahalnobis matching
require(MatchIt)
mah <- matchit(treated~age+education+re74+re75+black+hispanic+nodegree+married+u74+u75,
   distance="mahalanobis", data=LL)
mah
idx1 <- as.numeric(mah$match.matrix)
```

```
idx2 <- as.numeric(rownames(mah$match.matrix))
mah.idx <- match( c(idx1,idx2), rownames(LL))
#imbalance
L1.meas(LL$treated[mah.idx], LL[mah.idx,-c(1,9)], breaks=mybr)
eval.match(LL$treated[mah.idx], LL[mah.idx,-c(1,9)], breaks=mybr)
```

6.2 att: Example of ATT estimation from CEM output

Description

An example of ATT estimation from CEM output

Usage

```
att(obj, formula, data, model="lm", family="binomial")
```

Arguments

obj	a cem or multicem object
data	a single data.frame or a list of data.frame's in case of multicem
formula	formula type specification of model. See Details.
model	either lm or glm. See Details.
family	used if model is glm, otherwise ignored.

Details

Argument data must be a single data frame or a list of (mulitply imputed) data frames.

Argument model can be lm or glm if the outcome variable in the ATT estimation is, e.g., a binary outcome. If the outcome is y and the treatment variable is T, then a formula like y ~ T is enough to estimate the ATT: it is just the coefficient of T. User can add covariates to span any remaining imbalance after the match, such as y ~ T + age + sex, to adjust for variables age and sex.

In the case of multiply imputed datasets, the model is applied to each single matched data and the ATT and is the standard error estimated using the standard formulas for combining results of multiply imputed data.

Value

A matrix of estimates with their standard error, or a list in case of multicem.

Examples

```
 data(LL)

# cem match: automatic bin choice
mat <- cem(treatment="treated",data=LL, drop="re78")
mat$tab
mat$k2k

# ATT estimate
```

```
att(mat, re78~treated,  data=LL)

# reduce the match into k2k using euclidean distance within cem strata
mat2 <- k2k(mat, LL, "euclidean", 1)
mat2$tab
mat2$k2k

# ATT estimate after k2k
att(mat2, re78~treated, data=LL)

# using multiply imputated data
require(Amelia)

data(LL)
n <- dim(LL)[1]
k <- dim(LL)[2]

# we generate missing values in 30
# randomly in one colum per row
LL1 <- LL
idx <- sample(1:n, .3*n)
invisible(sapply(idx, function(x) LL1[x,sample(2:k,1)] <<- NA))

# we use Amelia for multiple imputation

imputed <- amelia(LL1)

mat <- multicem("treated", datalist=imputed[1:5], drop="re78")

out <- att(mat, re78 ~ treated, data=imputed[1:5])

str(out)
```

6.3 DW: Dehejia-Wahba dataset

Description

A subset of the Lalonde dataset (see cited reference).

Usage

```
data(DW)
```

Format

A data frame with 445 observations on the following 10 variables.

`treated` treated variable indicator

`age` age

`education` years of education

`black` race indicator variable

`married` marital status indicator variable

`nodegree` indicator variable of not possessing a degree

`re74` real earnings in 1974

`re75` real earnings in 1975

`re78` real earnings in 1978 (post treatment outcome)

`hispanic` ethnic indicator variable

`u74` unemployment in 1974 indicator variable

`u75` unemployment in 1975 indicator variable

Source

see references

References

Dehejia, R., Wahba, S. (1999) "Causal Effects in Nonexperimental Studies: Reevaluating the Evaluation of Training Programs," *Journal of the American Statistical Association*, 94, 1053-1062.

6.4 `eval.match`: Calculates several one dimensional imbalance measures

Description

Calculates several one dimensional imbalance measures for the original and matched data sets

Usage

```
eval.match(group, data, breaks = NULL, weights)
```

Arguments

`group`	the group variable
`data`	the data
`breaks`	a list of vectors of cutpoints used to calculate L1 measure. See Details.
`weights`	weights

Details

This function calculate several imbalance measures. For numeric variables the difference in means (under the column `statistics`, the difference in quantiles and the L1 measure is calculated. For categorical variables the L1 measure and the Chi-squared distance (under column `statistics`) is calculated.

If the `breaks` are not specified, the same approach as in `cem` is used. Please refer to `cem` help page. In this case, breaks are used to calculate the L1 measure.

Value

`value`	Table of imbalance measures

Author(s)

Stefano Iacus, Gary King, and Giusseppe Porro

References

Stefano Iacus, Gary King, Giusseppe Porro, "Matching for Casual Inference Without Balance Checking," http://gking.harvard.edu/files/abs/cem-abs.shtml

Examples

```
 data(LL)

mybr = list(re74=hist(LL$re74,plot=FALSE)$breaks,
 re75 = hist(LL$re75,plot=FALSE)$breaks,
 age = hist(LL$age,plot=FALSE)$breaks,
 education = hist(LL$education,plot=FALSE)$breaks)

L1.meas(LL$treated, LL[,-c(1,9)],breaks=mybr)
eval.match(LL$treated, LL[,-c(1,9)], breaks=mybr)

# cem match: automatic bin choice
mat <- cem(treatment="treated",data=LL, drop="re78")
mat$tab
cem1.idx <- which(mat$matched)
# imbalance
L1.meas(LL$treated[cem1.idx], LL[cem1.idx,-c(1,9)],breaks=mybr)
eval.match(LL$treated[cem1.idx], LL[cem1.idx,-c(1,9)], breaks=mybr)
eval.match(LL$treated[cem1.idx], LL[cem1.idx,-c(1,9)], breaks=mybr,weights=mat$w[cem1.idx])
```

6.5 k2k: Reduction to k2k Matching

Description

Reduces a CEM output to a k2k matching

Usage

```
k2k(obj, data, method=NULL, mpower=2, verbose=0)
```

Arguments

obj	an object as output from cem
data	the original data.frame used by cem
method	distance method to use in k2k matching. See Details.
mpower	power of the Minkowski distance. See Details.
verbose	controls level of verbosity. Default=0.

Details

This function transforms a typical cem matching solution to a k-to-k match, with k variable along strata: i.e., in each stratum generated by cem, the match is reduce to have the same number of treated and control units. (This option will delete some data that matched well, and thus likely increase the variance, but it means that subsequent analyses do not require weights.)

The user can choose a method (between 'euclidean', 'maximum', 'manhattan', 'canberra', 'binary' and 'minkowski') for nearest neighbor matching inside each cem strata. By default method is set to 'NULL', which means random matching inside cem strata. For the Minkowski distance the power can be specified via the argument mpower'. For more information on method != NULL, refer to dist help page.

After k2k the weights of each matched observation are set to unity.

Value

obj	a cem object

Author(s)

Stefano Iacus, Gary King, and Giusseppe Porro

References

Stefano Iacus, Gary King, Giusseppe Porro, "Matching for Casual Inference Without Balance Checking," http://gking.harvard.edu/files/abs/cem-abs.shtml

Examples

```
data(LL)

# cem match: automatic bin choice
mat <- cem(treatment="treated",data=LL, drop="re78")
mat$tab
mat$k2k

# ATT estimate
summary(lm(re78 ~ treated, data=LL, weights=mat$w))

# transform the match into k2k
mat2 <- k2k(mat, LL, "euclidean", 1)
mat2$tab
mat2$k2k

# ATT estimate after k2k
summary(lm(re78 ~ treated, data=LL, weights=mat2$w))
```

6.6 L1.meas: Evaluates L1 distance between multidimensional histograms

Description

Evaluates L1 distance between multidimensional histograms

Usage

```
L1.meas(group, data, breaks = NULL, weights)
```

Arguments

group	the group variable
data	the data
breaks	a list of vectors of cutpoints; if not specified, automatic choice will be made
weights	weights

Details

This function calculates the L1 distance on the k-dimensional histogram.

If the breaks are not specified, the same approach as in cem is used. Please refer to cem help page. In this case, breaks are used to calculate the L1 measure.

Value

value	the L1 measure

Author(s)

Stefano Iacus, Gary King, and Giusseppe Porro

References

Stefano Iacus, Gary King, Giusseppe Porro, "Matching for Casual Inference Without Balance Checking," http://gking.harvard.edu/files/abs/cem-abs.shtml

Examples

```
data(LL)
L1.meas(LL$treated,LL[,-c(1,9)])
```

6.7 LL: Lalonde dataset

Description

Lalonde experimental dataset (see cited reference).

Usage

```
data(LL)
```

Format

A data frame with 722 observations on the following 10 variables.

`treated` treatment variable indicator

`age` age

`education` years of education

`black` race indicator variable

`married` marital status indicator variable

`nodegree` indicator variable for not possessing a degree

`re74` real earnings in 1974

`re75` real earnings in 1975

`re78` real earnings in 1978 (post-treatment outcome)

`hispanic` ethnic indicator variable

`u74` unemployment in 1974 indicator variable

`u75` unemployment in 1975 indicator variable

Source

see references

References

Lalonde, R. (1986) "Evaluating the Econometric Evaluations of Training Programs," *American Economic Review*, 76, 604-620.

6.8 multicem: Coarsened Exact Matching for Multiply Imputed Data

Description

Implementation of Coarsened Exact Matching for Multiply Imputed Data

Usage

```
multicem(treatment = NULL, datalist, data = NULL, cutpoints = NULL, drop = NULL,
    eval.imbalance = FALSE, k2k = FALSE,  method=NULL, mpower=2, verbose = 0)
```

Arguments

treatment	character, name of the treatment variable
datalist	a list of imputed data.frame's
data	original data.frame with missing values
cutpoints	named list each describing the cutpoints for the variables (the names are variable names). Each list element is either a vector of cutpoints, a number of cutpoints, or a method for automatic bin contruction. See Details.
drop	a vector of variable names in the data frame to ignore during matching
eval.imbalance	boolean. See Details.
k2k	boolean, return k-to-k matching? Default = FALSE
method	distance method to use in k2k matching. See Details.
mpower	power of the Minkowski distance. See Details.
verbose	controls level of verbosity. Default=0.

Details

Argument datalist is a list of (multiply imputed) data frames. If data is not specified, the function cem is applied independently to each element of the list, resulting in separately matched data sets with different numbers of treated and control units.

When data is specified, each multiply imputed observation is assigned to the stratum in which it has been matched most frequently. In this case, the algorithm outputs the same matching solution for each multiply imputed data set (i.e., an observation, and the number of treated and control units matched, in one data set has the same meaning in all, and is the same for all)

All the remaining arguments are passed to cem as specified.

Value

An object of class `multicem`, i.e. a list of objects of class `cem`

Author(s)

Stefano Iacus, Gary King, and Giusseppe Porro

References

Stefano Iacus, Gary King, Giusseppe Porro, "Matching for Casual Inference Without Balance Checking," http://gking.harvard.edu/files/abs/cem-abs.shtml

Examples

```
require(Amelia)

data(LL)
n <- dim(LL)[1]
k <- dim(LL)[2]

set.seed(123)

LL1 <- LL
idx <- sample(1:n, .3*n)
invisible(sapply(idx, function(x) LL1[x,sample(2:k,1)] <<- NA))

imputed <- amelia(LL1,noms=c("black","hispanic","treated","married","nodegree","u74","u75"))

# without information on which observation has missing values
mat1 <- multicem("treated", datalist=imputed, drop="re78")
#str(mat1, max.lev=1)
mat1$match1$tab
mat1$match2$tab

# ATT estimation
out <- att(mat1, re78 ~ treated, data=imputed)

# with information about missingness
mat2 <- multicem("treated", datalist=imputed, drop="re78", data=LL1)
#str(mat2, max.lev=1)
mat2$match1$tab
mat2$match2$tab

# ATT estimation
out <- att(mat2, re78 ~ treated, data=imputed)
```

6.9 relax.cem: Diagnostic tool for CEM

Description

Diagnostic tools for CEM

Usage

```
relax.cem(obj, data, depth=1, verbose = 1, L1.breaks=NULL, plot=TRUE, fixed=NULL,
 shifts=NULL, minimal=NULL, use.coarsened=TRUE)
relax.plot(tab, group="1", max.terms=50, perc=.5, unique=FALSE, colors=TRUE)
```

Arguments

obj	an object of class cem.
data	the original data.
verbose	controls the level of verbosity.
L1.breaks	list of cutpoints for the calculation of the L1 measure.
plot	plot the solutions?
tab	the output table from relax.cem.
fixed	vector of variable names which will not be relaxed.
max.terms	plot only the last best results of relax.cem.
shifts	a vector of proportions of shifts.
minimal	the minimal number of intervals acceptable after relaxation. Should be a nameed list of positive integers.
group	character string denoting group id. Defaults to "1".
perc	only plot if percentage of matched units is greater than perc.
unique	only plot different solutions (in terms of matched units).
depth	if 1, relaxes up to dropping one var, if 2 relaxes (up to dropping) two vars, etc.
use.coarsened	used coarsened values for continuous variables.
colors	If TRUE each variable is plotted in a different colour.

Details

relax.cem starts from a cem solution (as given by cem) and tries several relaxed coarsenings on the variables. Coarsenings corresponds to dividing the support of each variable into a decreasing number of intervals of the same length (even if in the starting solution intervals are of different lengths). Because CEM is MIB, the number of matched units

increases as the number of intervals decrease. All variables are coarsened into `k` intervals along a sequence which starts from the original number of intervals and decreases to 10 intervals by 2, then continues from 10 down to 1 intervals by 1. If `minimal` is specified, variables are coarsened down to that minimal value.

To observe MIB property of CEM `use.coarsened` (default) should be set to `TRUE`; otherwise the coarsening of the continuous variable will be recalculated at each iteration and there is no guarantee of monotonicity.

`relax.cem` outputs a list of tables. Each table is named `Ggroup` where `group` is the id of the group. Each `Ggroup` table is ordered in increasing order of matched units of group `group`. Columns `PercGgroup` and `Ggroup` report percentage and absolute number of matched units for each `group`. Column `Relaxed` indicates which relaxation has been done, with something like `"V1(4), V3(5)"`, which means "variable `V1` has been split in 4 intervals of the same length and variable `V3` into five intervals". Thus, the number of intervals is reported in parenthases and if equal to 1 means that the corresponding variable is excluded from affecting the match (i.e. all observations are assigned to the same interval).

If `shifts` is not null, each coarsening is shifted accordingly (see `shift.cem` for additional details). In case of shifting "`S:`" appears in the labels.

The `relax.plot`, plot all the different relaxation in increasing order of number of treated units matched. For each coarsening it also reports the value of the L1 measure. The table generated by `relax.cem` may contain many entries. By default, only a portion of best coarsenings are plotted (option `max.terms`). In addition, the user can specify to plot the corasening for which at least a certain percentage of treated units have been matched (option `perc`, by default 50 In addition, of several different coarsenings which lead to the same number of treated units matched, the user can specify to plot only one of them using the option `unique` = `TRUE` (default).

Value

`tab` an invisible object containing the tabs

Author(s)

Stefano Iacus, Gary King, and Giusseppe Porro

References

Stefano Iacus, Gary King, Giusseppe Porro, "Matching for Casual Inference Without Balance Checking," http://gking.harvard.edu/files/abs/cem-abs.shtml

See Also

`cem`

Examples

```
data(LL)

mybr = list(re74=hist(LL$re74,plot=FALSE)$breaks,
  re75 = hist(LL$re75,plot=FALSE)$breaks,
  age = hist(LL$age,plot=FALSE)$breaks,
  education = hist(LL$education,plot=FALSE)$breaks)

mat <- cem(treatment="treated",data=LL, drop="re78")
mat$tab

tab <- relax.cem(mat, LL, L1.breaks=mybr, depth=1, plot=FALSE)

relax.plot(tab, group="1")
relax.plot(tab, group="1", unique=TRUE)
relax.plot(tab, group="1", perc=0.6)
relax.plot(tab, group="1", perc=0.6,unique=TRUE)

tab1 <- relax.cem(mat, LL, L1.breaks=mybr, depth=1, minimal=list(re74=6, age=3, education=3
tab2 <- relax.cem(mat, LL, L1.breaks=mybr, depth=1, minimal=list(re74=6, age=3, education=3
tab3 <- relax.cem(mat, LL, L1.breaks=mybr, depth=1, minimal=list(age=3, education=3), fixed=

# uncomment to run. Might be slow
# tab4 <- relax.cem(mat, LL, L1.breaks=mybr, depth=2, minimal=list(age=4, education=3,re75=6
# relax.plot(tab4)
# relax.plot(tab4, unique=TRUE)
# relax.plot(tab4, perc=0.7)
```

6.10 `shift.cem`: Diagnostic tool for CEM

Description

Diagnostic tools for CEM. Applies leftward and rightward shifts of the cutpoints.

Usage

```
shift.cem(obj, data, shifts=NULL, verbose=0, plot=TRUE)
```

Arguments

`obj`	and object of class cem
`data`	the original data
`shifts`	a vector of proportions of shifts
`verbose`	controls the level of verbosity
`plot`	whether to plot a graphic representation of the search

Details

For each variable, shift all the cutpoints left and right by `shifts` times the smallest epsilon of the coarsening. Shifting to the right produces a new cell on the left; shift to the left, adds a new cell to the coarsening on the right. Only positive proportions should be used; the algorithm will produce shifting on the left or on the right. The best shifting of the original cem match is produced as output, where best is defined in terms of the maximal total number of matched units `mT+mC` (see below).

By default, the function returns minimal information about the execution of the algorithm. By setting a value greater than 0 in option `verbose` more feedback on the process is returned.

Option `plot` = `TRUE` plots the number of treated units matched `mT`, the number of control units matched `mC`, and the sum `mT+mC`, as a function of the shifts.

Value

`tab`	an invisible object containing a new cem object

Author(s)

Stefano Iacus, Gary King, and Giusseppe Porro

References

Stefano Iacus, Gary King, Giusseppe Porro, "Matching for Casual Inference Without Balance Checking," http://gking.harvard.edu/files/abs/cem-abs.shtml

See Also

cem

Examples

```
data(LL)

m74 <- max(LL$re74, na.rm=TRUE)
s74 <- seq(0,m74,by=sd(LL$re74))
l74 <- length(s74)
if(max(s74) < m74) s74 <- c(s74, m74)

m75 <- max(LL$re75, na.rm=TRUE)
s75 <- seq(0,m75,by=sd(LL$re75))
l75 <- length(s75)
if(max(s75) < m75) s75 <- c(s75, m75)

mybr = list(re74=s74,
 re75 = s75,
 age = hist(LL$age,plot=FALSE)$breaks,
 education = hist(LL$education,plot=FALSE)$breaks)

mat <- cem(treatment="treated",data=LL, drop="re78",cut=mybr)
mat$tab

shift.cem(mat, data=LL, shifts=seq(0.01, 0.5, length=10), verb=1)
```

References

Ho, Daniel E., Kosuke Imai, Gary King and Elizabeth A. Stuart. Forthcoming. "MatchIt: Nonparametric Preprocessing for Parametric Causal Inference." *Journal of Statistical Software* . http://gking.harvard.edu/matchit.

Iacus, Stefano M., Gary King and Giuseppe Porro. 2008. "Matching for Causal Inference Without Balance Checking.". http://gking.harvard.edu/files/abs/cem-abs.shtml.