

Package ‘matchingMarkets’

January 30, 2016

Version 0.2-1

Depends R (>= 3.0.2)

Imports Rcpp (>= 0.11.2), lpSolve (>= 5.6.6), partitions, stats

LinkingTo Rcpp, RcppArmadillo, RcppProgress (>= 0.2)

Suggests knitr, testthat

VignetteBuilder knitr

Title Analysis of Stable Matchings

Author Thilo Klein

Maintainer Thilo Klein <thilo@klein.uk>

Description Implements structural estimators to correct for the sample selection bias from observed outcomes in matching markets. This includes one-sided matching of agents into groups as well as two-sided matching of students to schools. The package also contains R code for three matching algorithms: the deferred-acceptance (or Gale-Shapley) algorithm for stable marriage and college admissions problems, the top-trading-cycles algorithm for house allocation and a partitioning linear programme for the roommates problem.

URL <https://github.com/thiloklein/matchingMarkets>,
<http://matchingmarkets.r-forge.r-project.org>, <http://klein.uk>

BugReports <https://github.com/thiloklein/matchingMarkets/issues>

License GPL (>= 2) | file LICENSE

RoxygenNote 5.0.1

Repository CRAN

Repository/R-Forge/Project matchingmarkets

Repository/R-Forge/Revision 44

Repository/R-Forge/DateTimeStamp 2016-01-30 11:57:19

Date/Publication 2016-01-30 18:51:05

NeedsCompilation yes

R topics documented:

matchingMarkets-package	2
baac00	4
daa	5
khb	7
klein15a	8
klein15b	9
mce	11
mfx	12
plp	13
stabit	14
stabit2	19
stabsim	23
stabsim2	24
ttc	25
Index	27

matchingMarkets-package

An R package for the analysis of stable matchings.

Description

The matchingMarkets package contains R and C++ code for the estimation of structural models that correct for the sample selection bias of observed outcomes in matching markets.

Matching is concerned with who transacts with whom, and how. For example, who works at which job, which students go to which school, who forms a workgroup with whom, and so on.

The empirical analysis of matching markets is naturally subject to sample selection problems. If agents match assortatively on characteristics unobserved to the analyst but correlated with both the exogenous variable and the outcome of interest, regression estimates will generally be biased.

The matchingMarkets package comprises

1. *Bayes estimators.* The estimators implemented in function `stabit` and `stabit2` correct for the selection bias from endogenous matching.

The current package version provides solutions for two commonly observed matching processes: (i) the *group formation problem* with fixed group sizes and (ii) the *college admissions problem*. These processes determine which matches are observed – and which are not – and this is a sample selection problem.

2. *Post-estimation tools.* Function `mfx` computes marginal effects from coefficients in binary outcome and selection equations and `khb` implements the Karlson-Holm-Breen test for confounding due to sample selection.

3. *Design matrix generation.* The estimators are based on independent variables for all feasible, i.e., observed and counterfactual, matches in the market. Generating the characteristics of all feasible matches from individual-level data is a combinatorial problem. The `stabit` function has an argument `method="model.frame"` that returns a design matrix based on pre-specified transformations to generate counterfactual matches.
4. *Algorithms.* The package also contains several matching algorithms such as the deferred acceptance algorithm (`daa`) for **stable marriages** and **college admissions**, the top-trading-cycles algorithm (`ttc`) for **house allocation** and a partitioning linear program (`plp`) for the **stable room-mates problem**. These can be used to obtain stable matchings from simulated or real preference data.
5. *Data.* In addition to the `baac00` dataset from borrowing groups in Thailand's largest agricultural lending program, the package provides functions to simulate one's own data from matching markets. `stabsim` generates individual-level data and the `stabit` function has an argument `simulation` which generates group-level data and determines which groups are observed in equilibrium based on an underlying linear stochastic model.

Frequently Asked Questions

- *Why can I not use the classic Heckman correction?*

Estimators such as the Heckman (1979) correction (in package `sampleSelection`) or double selection models are inappropriate for this class of selection problems. To see this, note that a simple first stage discrete choice model assumes that an observed match reveals match partners' preferences over each other. In a matching market, however, agents can only choose from the set of partners who would be willing to form a match with them and we do not observe the players' relevant choice sets.

- *Do I need an instrumental variable to estimate the model?*

Short answer: No. Long answer: The characteristics of other agents in the market serve as the source of exogenous variation necessary to identify the model. The identifying exclusion restriction is that characteristics of all agents in the market affect the matching, i.e., who matches with whom, but it is only the characteristics of the match partners that affect the outcome of a particular match once it is formed. No additional instruments are required for identification (Sorensen, 2007).

- *What are the main assumptions underlying the estimator?*

The approach has certain limitations rooted in its restrictive economic assumptions.

1. The matching models are *complete information* models. That is, agents are assumed to have a complete knowledge of the qualities of other market participants.
2. The models are *static equilibrium* models. This implies that (i) the observed matching must be an equilibrium, i.e., no two agents would prefer to leave their current partners in order to form a new match (definition of pairwise stability), and (ii) the equilibrium must be unique for the likelihood function of the model to be well defined (Bresnahan and Reiss, 1991).
3. Uniqueness results can be obtained in two ways. First, as is common in the industrial organization literature, by imposing suitable *preference restrictions*. A suitable restriction on agents' preferences that guarantees a unique equilibrium is alignment (Pycia, 2012). In a group formation model, (pairwise) preference alignment states that any two agents who belong to the same groups must prefer the same group over the other. Second, by choosing a *market assignment* based on matching algorithms that produce a unique

stable matching, such as the well-studied Gale and Shapley (1962) deferred acceptance algorithm.

4. Finally, the models assume *bivariate normality* of the errors in selection and outcome equation. If that assumption fails, the estimator is generally inconsistent and can provide misleading inference in small samples.

How to cite this package

Whenever using this package, please cite as

Klein, T. (2015). `matchingMarkets`: Structural Estimator and Algorithms for the Analysis of Stable Matchings. R package version 0.2-1.

Author(s)

Thilo Klein

References

Bresnahan, T. and Reiss, P. (1991). Empirical models of discrete games. *Journal of Econometrics*, 48(1-2):57–81.

Gale, D. and Shapley, L.S. (1962). College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15.

Heckman, J. (1979). Sample selection bias as a specification error. *Econometrica*, 47(1):153–161.

Pycia, M. (2012). Stability and preference alignment in matching and coalition formation. *Econometrica*, 80(1):323–362.

Sorensen, M. (2007). How smart is smart money? A two-sided matching model of venture capital. *The Journal of Finance*, 62(6):2725–2762.

See Also

[sampleSelection](#)

baac00

Townsend Thai Project BAAC Annual Resurvey, 2000

Description

The `baac00` data frame contains data of 292 borrowers from Thailand’s largest agricultural lending program. These data are collected as part of the Townsend Thai Project Bank for Agriculture and Agricultural Cooperatives (BAAC) Annual Resurvey (Townsend, 2000). The 292 borrowers are nested within 68 groups and 39 markets. This nestedness makes the dataset particularly relevant for matching applications. A more complete discussion of the data is found in Ahlin (2009), Section 3, and Klein (2015a).

Usage

```
data(baac00)
```

Format

This data frame contains the following columns:

g.id group identifier.

m.id market identifier.

R repayment outcome: BAAC never raised interest rate as a penalty for late repayment.

pi success probability: measure of group members' project success probability.

wst worst year: indicator of economically worst year. 1:last year; 2:year before last year; 101-168:neither.

loan_size loan size: average loan size borrowed by the group.

loan_size2 loan size squared.

ingroup_agei log group age: log of number of years group has existed.

Source

Townsend, R. (2000). Townsend Thai Project Bank for Agriculture and Agricultural Cooperatives (BAAC) Annual Resurvey, 2000. Available at <http://hdl.handle.net/1902.1/12057>, *Murray Research Archive*.

References

Ahlin, C. (2009). Matching for credit: Risk and diversification in Thai microcredit groups. Working Paper 251, *Bureau for Research and Economic Analysis of Development*.

Klein, T. (2015a). *Does Anti-Diversification Pay? A One-Sided Matching Model of Microcredit*. *Cambridge Working Papers in Economics*, #1521.

 daa

Deferred Acceptance Algorithm for two-sided matching markets

Description

Finds the student-optimal matching in the **college admissions** problem or the men-optimal matching in the **stable marriage** problem. Uses the Gale-Shapley (1962) Deferred Acceptance Algorithm with student/male offer based on given or randomly generated preferences.

Usage

```
daa(nStudents = ncol(s.prefs), nColleges = ncol(c.prefs), nSlots = rep(1,
  nColleges), s.prefs = NULL, c.prefs = NULL)
```

Arguments

nStudents	integer indicating the number of students (in the college admissions problem) or men (in the stable marriage problem) in the market. Defaults to <code>ncol(s.prefs)</code> .
nColleges	integer indicating the number of colleges (in the college admissions problem) or women (in the stable marriage problem) in the market. Defaults to <code>ncol(c.prefs)</code> .
nSlots	vector of length <code>nColleges</code> indicating the number of places (i.e. quota) of each college. Defaults to <code>rep(1, nColleges)</code> for the marriage problem.
s.prefs	matrix of dimension <code>nColleges x nStudents</code> with the <code>j</code> th column containing student <code>j</code> 's ranking over colleges in decreasing order of preference (i.e. most preferred first).
c.prefs	matrix of dimension <code>nStudents x nColleges</code> with the <code>i</code> th column containing college <code>i</code> 's ranking over students in decreasing order of preference (i.e. most preferred first).

Value

daa returns a list with the following elements.

s.prefs	student-side preference matrix.
c.prefs	college-side preference matrix.
iterations	number of iterations required to find the stable matching.
matches	identifier of students/men assigned to colleges/women.
match.mat	matching matrix of dimension <code>nStudents x nColleges</code> .
singles	identifier of single (or unmatched) students/men.
edgelist	edgelist of matches

Minimum required arguments

daa requires the following combination of arguments, subject to the matching problem.

nStudents, nColleges Marriage problem with random preferences.

s.prefs, c.prefs Marriage problem with given preferences.

nStudents, nSlots College admissions problem with random preferences.

s.prefs, c.prefs, nSlots College admissions problem with given preferences.

Author(s)

Thilo Klein

References

Gale, D. and Shapley, L.S. (1962). College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15.

Oswald, F. (2013). Deferred Acceptance Algorithm with male offer. GitHub Gist, Available at <https://gist.github.com/floswald/1628636>

Examples

```
#####
## Marriage problem ##
#####

## 3 men, 2 women, random preferences:
daa(nStudents=3, nColleges=2)

## 3 men, 2 women, given preferences:
s.prefs <- matrix(c(1,2, 1,2, 1,2), 2,3)
c.prefs <- matrix(c(1,2,3, 1,2,3), 3,2)
daa(s.prefs=s.prefs, c.prefs=c.prefs)

#####
## College admission problem ##
#####

## 7 students, 2 colleges with 3 slots each, random preferences:
daa(nStudents=7, nSlots=c(3,3))

## 7 students, 2 colleges with 3 slots each, given preferences:
s.prefs <- matrix(c(1,2, 1,2, 1,2, 1,2, 1,2, 1,2, 1,2), 2,7)
c.prefs <- matrix(c(1,2,3,4,5,6,7, 1,2,3,4,5,6,7), 7,2)
daa(s.prefs=s.prefs, c.prefs=c.prefs, nSlots=c(3,3))
```

khh

*Karlson-Holm-Breen method for comparing probit coefficients***Description**

Significance test for confounding; that is, the difference between regression coefficients from same-sample nested logit and probit models. The test procedure follows Karlson et al (2012), Section 3.4.

Usage

```
khh(X, y, z)
```

Arguments

X	data frame comprising independent variables including confounding variable.
y	vector of dependent variable.
z	character string giving the name of the confounding variable in X.

Author(s)

Thilo Klein

References

Karlson, K.B., A. Holm and R. Breen (2012). Comparing regression coefficients between same-sample nested models using logit and probit: A new method. *Sociological Methodology*, 42(1):286–313.

Examples

```
## 1. load results from Klein (2015a)
data(klein15a)
M <- klein15a$model.list

## 2. extract variables
X <- do.call(rbind.data.frame, M$X)
eta <- c(klein15a$coefs$eta, rep(0, length(M$X)-length(M$W)))
X <- cbind(X,eta)
y <- unlist(M$R)

## 3. apply KHB method
khb(X=X, y=y, z="eta")
```

klein15a

MCMC results in Klein (2015a).

Description

MCMC results in Klein (2015a).

Usage

```
data(klein15a)
```

Format

This list contains the following elements:

model.list .

coefs .

References

Klein, T. (2015a). *Does Anti-Diversification Pay? A One-Sided Matching Model of Microcredit*. *Cambridge Working Papers in Economics*, #1521.

klein15b

*Results of Monte Carlo Simulations in Klein (2015b).***Description**

Results of Monte Carlo Simulations in Klein (2015b) for 40 two-group markets.

Usage

```
data(klein15b)
```

Format

This list contains the following elements:

exp.5.5.ols Benchmark study, OLS: coefficient estimates for 40 markets with groups of 5. Data for all 5 group members is observed.

exp.5.5.ntu Benchmark study, structural model.

exp.6.5.ols Experiment 1, OLS: coefficient estimates for 40 markets with groups of 6. Only Data for 5 group members is observed.

exp.6.5.ntu Experiment 1, structural model.

exp.6.6.ols Experiment 2, OLS: coefficient estimates for 40 markets with groups of 6. Data for all 6 group members is observed but only a random sample of 250 of the 922 counterfactual groups is used in the analysis.

exp.6.6.ntu Experiment 2, structural model.

References

Klein, T. (2015a). *Does Anti-Diversification Pay? A One-Sided Matching Model of Microcredit*. *Cambridge Working Papers in Economics*, #1521.

Klein, T. (2015b). *Analysis of stable matchings in R: Package matchingMarkets*. *Vignette to R package matchingMarkets*, The Comprehensive R Archive Network.

Examples

```
## Not run:
#####
## Modes of posterior distributions ##
#####

## load data
data(klein15b)

## define function to obtain the mode
mode <- function(x){
  d <- density(x,bw="SJ")
  formatC(round(d$x[which.max(d$y)], 3), format='f', digits=3)
```

```

}

## Benchmark study
apply(klein15b$exp.5.5.ntu, 2, mode)
apply(klein15b$exp.5.5.ols, 2, mode)

## Experiment 1
apply(klein15b$exp.6.5.ntu, 2, mode)
apply(klein15b$exp.6.5.ols, 2, mode)

## Experiment 2
apply(klein15b$exp.6.6.ntu, 2, mode)
apply(klein15b$exp.6.6.ols, 2, mode)

#####
## Plot of posterior distributions ##
#####

## load data
data(klein15b)

par(mfrow=c(3,3))
tpe <- c(rep("Benchmark",2), rep("Experiment 1",2), rep("Experiment 2",2))
par(mar=c(5.1,4.6,0.8,2.1))

for(i in seq(1,length(klein15b)-1,2)){
  ntu <- klein15b[[i]]
  ols <- klein15b[[i+1]]

  ntu <- ntu[,colnames(ntu)]
  ols <- ols[,colnames(ols) == "beta.wst.ieq"]

  plot(density(ntu[,1]), xlab=expression(hat(alpha)),
       ylab="density", main="", axes=FALSE, xlim=c(-1,2))
  axis(2,lwd=2,cex.axis=0.8); axis(1,lwd=2,cex.axis=0.8)
  legend("topleft","Struct.",lty=1,bty="n")
  abline(v=1, lty=3)

  plot(density(ntu[,2]), xlab=expression(hat(beta)),
       ylab="density", main=tpe[i], axes=FALSE)
  axis(2,lwd=2,cex.axis=0.8); axis(1,lwd=2,cex.axis=0.8)
  points(density(ols), type="l", lty=2)
  legend("topright",c("Struct.", "OLS"),lty=c(1,2),bty="n")
  abline(v=-1, lty=3)

  plot(density(ntu[,3]), xlab=expression(hat(delta)),
       ylab="density", main="", axes=FALSE)
  axis(2,lwd=2,cex.axis=0.8); axis(1,lwd=2,cex.axis=0.8)
  legend("topleft","Struct.",lty=1,bty="n")
  abline(v=0.5, lty=3)
}

## End(Not run)

```

mce *MC Experiments*

Description

MC Experiments

Usage

```
mce(seed, niter, N, n, m, type, method)
```

Arguments

seed	seed value for Monte Carlo Experiment
niter	number of draws in estimation
N	group size (population)
n	group size (sample)
m	number of markets
type	type of the MC Experiment. Either <code>group.members</code> for randomly sampled group members or <code>counterfactual.groups</code> for randomly sampled number of counterfactual (or feasible) groups in selection equation (capped at <code>limit.max.combs=250</code>)
method	either <code>group.members</code> or <code>counterfactual.groups</code>

Author(s)

Thilo Klein

Examples

```
## Not run:
## 1. Set parameters
mciter <- 2 #500
niter <- 10 #400000
nodes <- 4

## 2. Setup parallel backend to use 4 processors
library(foreach); library(doSNOW)
cl <- makeCluster(4); registerDoSNOW(cl)

## 3. Define foreach loop function
mce.add <- function(mciter, niter, N, n, m, type, method){
  h <- foreach(i=1:mciter) %dopar% {
    library(matchingMarkets)
    mce(seed=i,niter, N, n, m, type, method)
  }
  do.call(rbind, h)
}
```

```

## 4. Run simulations:

## 4-a. Benchmark study
exp.5.5.ols <- mce.add(mciter=mciter, niter=niter, N=5, n=5, m=40,
                      type="group.members", method="outcome")
exp.5.5.ntu <- mce.add(mciter=mciter, niter=niter, N=5, n=5, m=40,
                      type="group.members", method="NTU")

## 4-b. Experiment 1: randomly sampled group members
exp.6.5.ols <- mce.add(mciter=mciter, niter=niter, N=6, n=5, m=40,
                      type="group.members", method="outcome")
exp.6.5.ntu <- mce.add(mciter=mciter, niter=niter, N=6, n=5, m=40,
                      type="group.members", method="NTU")

## 4-c. Experiment 2: randomly sampled counterfactual groups
exp.6.6.ols <- mce.add(mciter=mciter, niter=niter, N=6, n=6, m=40,
                      type="counterfactual.groups", method="outcome")
exp.6.6.ntu <- mce.add(mciter=mciter, niter=niter, N=6, n=6, m=40,
                      type="counterfactual.groups", method="NTU")

## 5. Stop parallel backend
stopCluster(cl)

## End(Not run)

```

mfx

Marginal effects for probit and matching models

Description

Marginal effects from regression coefficients for probit and matching models.

Usage

```
mfx(m, toLatex = FALSE)
```

Arguments

m	an object returned from functions <code>stabit</code> or <code>stabit2</code> .
toLatex	logical: if TRUE the result tables are printed in Latex format. The default setting is FALSE.

Author(s)

Thilo Klein

References

Klein, T. (2015a). *Does Anti-Diversification Pay? A One-Sided Matching Model of Microcredit*. *Cambridge Working Papers in Economics*, #1521.

Examples

```
## 1. load results from Klein (2015a)
data(klein15a)

## 2. apply mfx function and print results
mfx(m=klein15a)
```

plp

Partitioning Linear Programme for the stable roommates problem

Description

Finds the stable matching in the **stable roommates problem** with transferable utility. Uses the Partitioning Linear Programme formulated in Quint (1991).

Usage

```
plp(V = NULL, N = NULL)
```

Arguments

V	valuation matrix of dimension $N \times N$ that gives row-players valuation over column players (or vice versa).
N	integer (divisible by 2) that gives the number of players in the market.

Value

plp returns a list with the following items.

Valuation.matrix	input values of V.
Assignment.matrix	upper triangular matrix of dimension $N \times N$ with entries of 1 for equilibrium pairs and 0 otherwise.
Equilibrium.groups	matrix that gives the $N/2$ equilibrium pairs and equilibrium partners' mutual valuations.

Author(s)

Thilo Klein

References

Quint, T. (1991). Necessary and sufficient conditions for balancedness in partitioning games. *Mathematical Social Sciences*, 22(1):87–91.

Examples

```
## Roommate problem with 10 players, transferable utility and random preferences:
plp(N=10)

## Roommate problem with 10 players, transferable utility and given preferences:
V <- matrix(rep(1:10, 10), 10, 10)
plp(V=V)
```

stabit

Structural Matching Model to correct for sample selection bias in one-sided matching markets

Description

The function provides a Gibbs sampler for a structural matching model that corrects for sample selection bias when the selection process is a one-sided matching game; that is, group/coalition formation.

The input is individual-level data of all group members from one-sided matching markets; that is, from group/coalition formation games.

In a first step, the function generates a model matrix with characteristics of *all feasible* groups of the same size as the observed groups in the market.

For example, in the stable roommates problem with $n = 4$ students $\{1, 2, 3, 4\}$ sorting into groups of 2, we have $\binom{4}{2} = 6$ feasible groups: (1,2)(3,4) (1,3)(2,4) (1,4)(2,3).

In the group formation problem with $n = 6$ students $\{1, 2, 3, 4, 5, 6\}$ sorting into groups of 3, we have $\binom{6}{3} = 20$ feasible groups. For the same students sorting into groups of sizes 2 and 4, we have $\binom{6}{2} + \binom{6}{4} = 30$ feasible groups.

The structural model consists of a selection and an outcome equation. The *Selection Equation* determines which matches are observed ($D = 1$) and which are not ($D = 0$).

$$\begin{aligned} D &= 1[V \in \Gamma] \\ V &= W\alpha + \eta \end{aligned}$$

Here, V is a vector of latent valuations of *all feasible* matches, ie observed and unobserved, and $1[\cdot]$ is the Iverson bracket. A match is observed if its match valuation is in the set of valuations Γ that satisfy the equilibrium condition (see Klein, 2015a). This condition differs for matching games with transferable and non-transferable utility and can be specified using the method argument. The match valuation V is a linear function of W , a matrix of characteristics for *all feasible* groups, and η , a vector of random errors. α is a parameter vector to be estimated.

The *Outcome Equation* determines the outcome for *observed* matches. The dependent variable can either be continuous or binary, dependent on the value of the binary argument. In the binary case, the dependent variable R is determined by a threshold rule for the latent variable Y .

$$\begin{aligned} R &= 1[Y > c] \\ Y &= X\beta + \epsilon \end{aligned}$$

Here, Y is a linear function of X , a matrix of characteristics for *observed* matches, and ϵ , a vector of random errors. β is a parameter vector to be estimated.

The structural model imposes a linear relationship between the error terms of both equations as $\epsilon = \delta\eta + \xi$, where ξ is a vector of random errors and δ is the covariance parameter to be estimated. If δ were zero, the marginal distributions of ϵ and η would be independent and the selection problem would vanish. That is, the observed outcomes would be a random sample from the population of interest.

Usage

```
stabit(x, m.id = "m.id", g.id = "g.id", R = "R", selection = NULL,
       outcome = NULL, roommates = FALSE, simulation = "none", seed = 123,
       max.combs = Inf, method = "NTU", binary = FALSE, offsetOut = 0,
       offsetSel = 0, marketFE = FALSE, censored = 0, gPrior = FALSE,
       dropOnes = FALSE, interOut = 0, interSel = 0, standardize = 0,
       niter = 10)
```

Arguments

<code>x</code>	data frame with individual-level characteristics of all group members including market- and group-identifiers.
<code>m.id</code>	character string giving the name of the market identifier variable. Defaults to "m.id".
<code>g.id</code>	character string giving the name of the group identifier variable. Defaults to "g.id".
<code>R</code>	dependent variable in outcome equation. Defaults to "R".
<code>selection</code>	list containing variables and pertaining operators in the selection equation. The format is <code>operation = "variable"</code> . See the Details and Examples sections.
<code>outcome</code>	list containing variables and pertaining operators in the outcome equation. The format is <code>operation = "variable"</code> . See the Details and Examples sections.
<code>roommates</code>	logical: if TRUE data is assumed to come from a roommate game. This means that groups are of size two and the model matrix is prepared for individual-level analysis (peer-effects estimation). If FALSE (which is the default) data is assumed to come from a group/coalition formation game and the model matrix is prepared for group-level analysis.
<code>simulation</code>	should the values of dependent variables in selection and outcome equations be simulated? Options are "none" for no simulation, "NTU" for non-transferable utility matching, "TU" for transferable utility or "random" for random matching of individuals to groups. Simulation settings are (i) all model coefficients set to $\alpha=\beta=1$; (ii) covariance between error terms $\delta=0.5$; (iii) error terms η and ξ are draws from a standard normal distribution.

seed	integer setting the state for random number generation if simulation=TRUE.
max.combs	integer (divisible by two) giving the maximum number of feasible groups to be used for generating group-level characteristics.
method	estimation method to be used. Either "NTU" or "TU" for selection correction using non-transferable or transferable utility matching as selection rule; "outcome" for estimation of the outcome equation only; or "model.frame" for no estimation.
binary	logical: if TRUE outcome variable is taken to be binary; if FALSE outcome variable is taken to be continuous.
offsetOut	vector of integers indicating the indices of columns in X for which coefficients should be forced to 1. Use 0 for none.
offsetSel	vector of integers indicating the indices of columns in W for which coefficients should be forced to 1. Use 0 for none.
marketFE	logical: if TRUE market-level fixed effects are used in outcome equation; if FALSE no market fixed effects are used.
censored	draws of the delta parameter that estimates the covariation between the error terms in selection and outcome equation are 0:not censored, 1:censored from below, 2:censored from above.
gPrior	logical: if TRUE the g-prior (Zellner, 1986) is used for the variance-covariance matrix.
dropOnes	logical: if TRUE one-group-markets are excluded from estimation.
interOut	two-column matrix indicating the indices of columns in X that should be interacted in estimation. Use 0 for none.
interSel	two-column matrix indicating the indices of columns in W that should be interacted in estimation. Use 0 for none.
standardize	numeric: if standardize>0 the independent variables will be standardized by dividing by standardize times their standard deviation. Defaults to no standardization standardize=0.
niter	number of iterations to use for the Gibbs sampler.

Details

Operators for variable transformations in selection and outcome arguments.

add sum over all group members and divide by group size.

int sum over all possible two-way interactions $x * y$ of group members and divide by the number of those, given by choose(n, 2).

ieq sum over all possible two-way equality assertions $1[x = y]$ and divide by the number of those.

ive sum over all possible two-way interactions of vectors of variables of group members and divide by number of those.

inv ...

dst sum over all possible two-way distances between players and divide by number of those, where distance is defined as $e^{-|x-y|}$.

sel for roommates=TRUE only: variable for individual (for peer effects estimation).

oth for roommates=TRUE only: variable for other in the group (for peer effects estimation).

Value

stabit returns a list with the following items.

model.list
 model.frame
 draws
 coefs

Values of model.list

D vector that indicates – for all feasible groups in the market – whether a group is observed in the data $D=1$ or not $D=0$.

R list of group-level outcome vectors for equilibrium groups.

W list with data.frame $W[[t]][G,]$ containing characteristics of group G in market t (all feasible groups).

X list with data.frame $X[[t]][G,]$ containing characteristics of group G in market t (equilibrium groups only).

V vector of group valuations for all feasible groups in the market.

P vector that gives for each group the index of the group comprised of residual individuals in the market (for 2-group markets).

epsilon if simulation!="none", the errors in the outcome equation, given by $\delta \cdot \eta + \xi$.

eta if simulation!="none", the standard normally distributed errors in the selection equation.

xi if simulation!="none", the standard normally distributed component of the errors in the selection equation that is independent of eta.

combs partitions matrix that gives all feasible partitions of the market into groups of the observed sizes.

E matrix that gives the indices of equilibrium group members for each group in the market. Only differs from the first two rows in combs if simulation!="none".

sigmasquareximean variance estimate of the error term xi in the outcome equation.

Values of model.frame

SEL data frame comprising variables in selection equation and number of observations equal to the number of feasible groups.

OUT data frame comprising variables in outcome equation and number of observations equal to the number of equilibrium groups.

Values of draws

alphadraws matrix of dimension $ncol(W) \times niter$ comprising all parameter draws for the selection equation.

betadraws matrix of dimension $ncol(X) \times niter$ comprising all parameter draws for the outcome equation.

deltadraws vector of length niter comprising all draws for the delta parameter.

sigmasquarexidraws .

Values of coefs

eta vector containing the mean of all eta draws for each observed group.
 alphavcov variance-covariance matrix of draws in alphadraws.
 betavcov variance-covariance matrix of draws in betadraws.
 alpha matrix comprising the coefficient estimates of alpha and their standard errors.
 beta matrix comprising the coefficient estimates of beta and their standard errors.
 delta coefficient estimate of delta and its standard error.
 sigmasquarexi variance estimate of the error term xi in the outcome equation and its standard error.

Author(s)

Thilo Klein

References

Klein, T. (2015a). *Does Anti-Diversification Pay? A One-Sided Matching Model of Microcredit*. *Cambridge Working Papers in Economics*, #1521.

Zellner, A. (1986). *On assessing prior distributions and Bayesian regression analysis with g-prior distributions*, volume 6, pages 233–243. North-Holland, Amsterdam.

Examples

```
#####
## MODEL FRAMES (method="model.frame") ##
#####

## --- ROOMMATES GAME ---
## 1. Simulate one-sided matching data for 3 markets (m=3) with 3 groups
## per market (gpm=3) and 2 individuals per group (ind=2)
idata <- stabsim(m=3, ind=2, gpm=3)
## 2. Obtain the model frame
s1 <- stabit(x=idata, selection = list(add="pi", ieq="wst"),
            outcome = list(add="pi", ieq="wst"),
            method="model.frame", simulation="TU", roommates=TRUE)

## --- GROUP/COALITION FORMATION (I) ---
## 1. Simulate one-sided matching data for 3 markets (m=3) with 2 groups
## per market (gpm=2) and 2 to 4 individuals per group (ind=2:4)
idata <- stabsim(m=3, ind=2:4, gpm=2)
## 2. Obtain the model frame
s2 <- stabit(x=idata, selection = list(add="pi", ieq="wst"),
            outcome = list(add="pi", ieq="wst"),
            method="model.frame", simulation="NTU", roommates=FALSE)

## --- GROUP/COALITION FORMATION (II) ---
## Not run:
## 1. Load baac00 data from the Townsend Thai project
data(baac00)
```

```

## 2. Obtain the model frame
s3 <- stabit(x=baac00, selection = list(add="pi", int="pi", ieq="wst", ive="occ"),
  outcome = list(add="pi", int="pi", ieq="wst", ive="occ",
    add=c("loan_size", "loan_size2", "lngroup_agei")),
  method="model.frame", simulation="none")

## End(Not run)

#####
## ESTIMATION (method="NTU") ##
#####

## Not run:
## --- SIMULATED EXAMPLE ---
## 1. Simulate one-sided matching data for 3 markets (m=3) with 2 groups
## per market (gpm=2) and 2 to 4 individuals per group (ind=2:4)
idata <- stabsim(m=3, ind=2:4, gpm=2)
## 2. Run Gibbs sampler
fit1 <- stabit(x=idata, selection = list(add="pi",ieq="wst"),
  outcome = list(add="pi",ieq="wst"),
  method="NTU", simulation="NTU", binary=FALSE, niter=2000)
## 3. Get results
names(fit1)

## --- REPLICATION, Klein (2015a) ---
## 1. Load data
data(baac00); head(baac00)
## 2. standardise variables
baac00$pi <- baac00$pi + (1-baac00$pi)*0.5
baac00$loan_size <- baac00$loan_size/sd(baac00$loan_size)
baac00$loan_size2 <- baac00$loan_size^2
baac00$lngroup_agei <- baac00$lngroup_agei/sd(baac00$lngroup_agei)
## 3. Run Gibbs sampler
klein15a <- stabit(x=baac00, selection = list(inv="pi",ieq="wst"),
  outcome = list(add="pi",inv="pi",ieq="wst",
    add=c("loan_size", "loan_size2", "lngroup_agei")), offsetOut=1,
  method="NTU", binary=TRUE, gPrior=TRUE, marketFE=TRUE, niter=800000)
## 4. Get results
mfx(klein15a)

## End(Not run)

```

stabit2

Structural Matching Model to correct for sample selection bias in two-sided matching markets

Description

The function provides a Gibbs sampler for a structural matching model that corrects for sample selection bias when the selection process is a two-sided matching game; i.e., a matching of students to colleges.

The structural model consists of a selection and an outcome equation. The *Selection Equation* determines which matches are observed ($D = 1$) and which are not ($D = 0$).

$$\begin{aligned} D &= 1[V \in \Gamma] \\ V &= W\alpha + \eta \end{aligned}$$

Here, V is a vector of latent valuations of *all feasible* matches, ie observed and unobserved, and $1[\cdot]$ is the Iverson bracket. A match is observed if its match valuation is in the set of valuations Γ that satisfy the equilibrium condition (see Sorensen, 2007). The match valuation V is a linear function of W , a matrix of characteristics for *all feasible* groups, and η , a vector of random errors. α is a parameter vector to be estimated.

The *Outcome Equation* determines the outcome for *observed* matches. The dependent variable can either be continuous or binary, dependent on the value of the binary argument. In the binary case, the dependent variable R is determined by a threshold rule for the latent variable Y .

$$\begin{aligned} R &= 1[Y > c] \\ Y &= X\beta + \epsilon \end{aligned}$$

Here, Y is a linear function of X , a matrix of characteristics for *observed* matches, and ϵ , a vector of random errors. β is a parameter vector to be estimated.

The structural model imposes a linear relationship between the error terms of both equations as $\epsilon = \delta\eta + \xi$, where ξ is a vector of random errors and δ is the covariance parameter to be estimated. If δ were zero, the marginal distributions of ϵ and η would be independent and the selection problem would vanish. That is, the observed outcomes would be a random sample from the population of interest.

Usage

```
stabit2(OUT, SEL = NULL, colleges = NULL, students = NULL,
  m.id = "m.id", c.id = "c.id", s.id = "s.id", outcome,
  selection = NULL, selection.student = NULL, selection.college = NULL,
  binary = FALSE, niter, gPrior = FALSE, censored = 1, seed = 123)
```

Arguments

OUT	data frame with characteristics of all observed matches, including market identifier <code>m.id</code> , college identifier <code>c.id</code> and student identifier <code>s.id</code> .
SEL	optional: data frame with characteristics of all observed and unobserved matches, including market identifier <code>m.id</code> , college identifier <code>c.id</code> and student identifier <code>s.id</code> .
colleges	character vector of variable names for college characteristics. These variables carry the same value for any college.
students	character vector of variable names for student characteristics. These variables carry the same value for any student.
<code>m.id</code>	character name of the market identifier variable. Defaults to <code>"m.id"</code> .
<code>c.id</code>	character name of the college identifier variable. Defaults to <code>"c.id"</code> .
<code>s.id</code>	character name of the student identifier variable. Defaults to <code>"s.id"</code> .

outcome	formula for match outcomes.
selection	formula for match valuations.
selection.student	formula for match valuations of students. This argument is ignored when selection is provided.
selection.college	formula for match valuations of colleges. This argument is ignored when selection is provided.
binary	logical: if TRUE outcome variable is taken to be binary; if FALSE outcome variable is taken to be continuous.
niter	number of iterations to use for the Gibbs sampler.
gPrior	logical: if TRUE the g-prior (Zellner, 1986) is used for the variance-covariance matrix. (Not yet implemented)
censored	draws of the delta parameter that estimates the covariation between the error terms in selection and outcome equation are 0:not censored, 1:censored from below, 2:censored from above.
seed	integer setting the state for MCMC draws.

Value

stabit returns a list with the following items.

```
model.list
draws
coefs
```

Author(s)

Thilo Klein

References

Sorensen, M. (2007). How Smart is Smart Money? A Two-Sided Matching Model of Venture Capital. *Journal of Finance*, 62 (6): 2725-2762.

Examples

```
## --- SIMULATED EXAMPLE ---
## Not run:
## 1. Simulate two-sided matching data for 20 markets (m=20) with 100 students
##   (nStudents=100) per market and 20 colleges with quotas of 5 students, each
##   (nSlots=rep(5,20)).

xdata <- stabsim2(m=20, nStudents=100, nSlots=rep(5,20),
  colleges = "c1",
  students = "s1",
  outcome = ~ c1:s1 + eta +nu,
  selection = ~ -1 + c1:s1 + eta
```

```

)
head(xdata$OUT)

## 2-a. Bias from sorting
lm1 <- lm(y ~ c1:s1, data=xdata$OUT)
summary(lm1)

## 2-b. Cause of the bias
with(xdata$OUT, cor(c1*s1, eta))

## 2-c. Correction for sorting bias
lm2a <- lm(V ~ -1 + c1:s1, data=xdata$SEL); summary(lm2a)
etahat <- lm2a$residuals[xdata$SEL$D==1]

lm2b <- lm(y ~ c1:s1 + etahat, data=xdata$OUT)
summary(lm2b)

## 3. Correction for sorting bias when match valuation V is unobserved

## 3-a. Run Gibbs sampler (when SEL is given)
fit2 <- stabit2(OUT = xdata$OUT,
               SEL = xdata$SEL,
               outcome = y ~ c1:s1,
               selection = ~ -1 + c1:s1,
               niter=1000
              )

## 3-b. Run Gibbs sampler (when SEL is not given)
fit2 <- stabit2(OUT = xdata$OUT,
               colleges = "c1",
               students = "s1",
               outcome = y ~ c1:s1,
               selection = ~ -1 + c1:s1,
               niter=1000
              )

## 4-a. Get marginal effects (for linear model)
fit2$coefs

## 4-b. Get marginal effects (for probit)
mfx(fit2)

## 5. Plot MCMC draws for coefficients
plot(fit2$draws$alphadraws[1,], type="l")
plot(fit2$draws$betadraws[1,], type="l")
plot(fit2$draws$deltadraws[1,], type="l")

## 6. Obtain the model list used in estimation
head(fit2$model.list)

```

```
## End(Not run)
```

```
stabsim Simulate individual-level data for one-sided matching markets
```

Description

Simulate individual-level data for one-sided matching markets.

Usage

```
stabsim(m, ind, seed = 123, singles = NULL, gpm = 2)
```

Arguments

<code>m</code>	integer indicating the number of markets to be simulated.
<code>ind</code>	integer (or vector) indicating the number of individuals per group.
<code>seed</code>	integer setting the state for random number generation. Defaults to <code>set.seed(123)</code> .
<code>singles</code>	integer giving the number of one-group markets.
<code>gpm</code>	integer giving the number of groups per market.

Value

`stabsim` returns a data frame with the randomly generated variables mimicking those in dataset [baac00](#).

<code>m.id</code>	categorical: market identifier.
<code>g.id</code>	categorical: group identifier.
<code>wst</code>	binary: indicator taking the value 1 if last year was worse than the year before; 0 otherwise.
<code>R</code>	NA: group outcome is not simulated. It can be obtained using the simulation argument in function <code>stabit</code> .

Author(s)

Thilo Klein

Examples

```
## Coalitions [gpm := 2 !]
## Simulate one-sided matching data for 4 markets (m=4) with 2 groups
## per market (gpm=2) and 2 to 4 individuals per group (ind=2:4)
idata <- stabsim(m=4, ind=2:4, seed=124, singles=2, gpm=2)

## Rommmates [ind := 2 !]
## Simulate one-sided matching data for 3 markets (m=3) with 3 groups
## per market (gpm=3) and 2 individuals per group (ind=2)
idata <- stabsim(m=3, ind=2, seed=124, gpm=3)
```

stabsim2

Simulate data for two-sided matching markets

Description

Simulate data for two-sided matching markets. In the simulation for the Sorensen (2007) with one selection equation (`selection`), an equal sharing rule of $\lambda = 0.5$ is used.

Usage

```
stabsim2(m, nStudents, nColleges = length(nSlots), nSlots, colleges, students,
outcome, selection = NULL, selection.student = NULL,
selection.college = NULL, seed = 123)
```

Arguments

<code>m</code>	integer indicating the number of markets to be simulated.
<code>nStudents</code>	integer indicating the number of students per market.
<code>nColleges</code>	integer indicating the number of colleges per market.
<code>nSlots</code>	vector of length <code>nColleges</code> indicating the number of places at each college, i.e. the college's quota.
<code>colleges</code>	character vector of variable names for college characteristics. These variables carry the same value for any college.
<code>students</code>	character vector of variable names for student characteristics. These variables carry the same value for any student.
<code>outcome</code>	formula for match outcomes.
<code>selection</code>	formula for match valuations.
<code>selection.student</code>	formula for match valuations of students. This argument is ignored when <code>selection</code> is provided.
<code>selection.college</code>	formula for match valuations of colleges. This argument is ignored when <code>selection</code> is provided.
<code>seed</code>	integer setting the state for random number generation. Defaults to <code>set.seed(123)</code> .

Value

`stabsim2` returns a list with the following items.

```
OUT
SEL
SELc
SELS
```


Author(s)

Thilo Klein

Examples

```
## Not run:
## Simulate two-sided matching data for 2 markets (m=2) with 10 students
## (nStudents=10) per market and 3 colleges (nColleges=3) with quotas of
## 2, 3, and 5 students, respectively.

xdata <- stabsim2(m=2, nStudents=10, nSlots=c(2,3,5),
  colleges = "c1",
  students = "s1",
  outcome = ~ c1:s1 + eta + nu,
  selection = ~ -1 + c1:s1 + eta
)
head(xdata$OUT)
head(xdata$SEL)

## End(Not run)
```

ttc

Top-Trading-Cycles Algorithm for the house allocation problem

Description

Finds the stable matching in the **house allocation problem** with existing tenants. Uses the Top-Trading-Cycles Algorithm proposed in Abdulkadiroglu and Sonmez (1999).

Usage

```
ttc(P = NULL, X = NULL)
```

Arguments

P	list of individuals' preference rankings over objects.
X	2-column-matrix of objects (obj) and their owners (ind).

Value

ttc returns a 2-column matrix of the stable matching solution for the housing market problem based on the Top-Trading-Cycles algorithm.

Author(s)

Thilo Klein

References

Abdulkadiroglu, A. and Sonmez, T. (1999). House Allocation with Existing Tenants. *Journal of Economic Theory*, 88(2):233–260.

Examples

```
## generate list of individuals' preference rankings over objects
P <- list()
P[[1]] <- c(2,5,1,4,3) # individual 1
P[[2]] <- c(1,5,4,3,2) # individual 2
P[[3]] <- c(2,1,4,3,5) # individual 3
P[[4]] <- c(2,4,3,1,5) # individual 4
P[[5]] <- c(4,3,1,2,5); P # individual 5

## generate 2-column-matrix of objects ('obj') and their owners ('ind')
X <- data.frame(ind=1:5, obj=1:5); X

## find assignment based on TTC algorithm
ttc(P=P,X=X)
```

Index

- *Topic **algorithms**
 - daa, [5](#)
 - plp, [13](#)
 - ttc, [25](#)
- *Topic **datasets**
 - baac00, [4](#)
 - klein15a, [8](#)
 - klein15b, [9](#)
- *Topic **generate**
 - stabsim, [23](#)
 - stabsim2, [24](#)
- *Topic **package**
 - matchingMarkets-package, [2](#)
- *Topic **regression**
 - stabit, [14](#)
 - stabit2, [19](#)
- *Topic **summary**
 - khb, [7](#)
 - mfx, [12](#)

[baac00](#), [3](#), [4](#), [23](#)

[daa](#), [3](#), [5](#)

[khb](#), [2](#), [7](#)

[klein15a](#), [8](#)

[klein15b](#), [9](#)

[matchingMarkets](#)
([matchingMarkets-package](#)), [2](#)

[matchingMarkets-package](#), [2](#)

[mce](#), [11](#)

[mfx](#), [2](#), [12](#)

[plp](#), [3](#), [13](#)

[stabit](#), [2](#), [3](#), [14](#)

[stabit2](#), [2](#), [19](#)

[stabitCpp \(stabit\)](#), [14](#)

[stabitCpp2 \(stabit2\)](#), [19](#)

[stabsim](#), [3](#), [23](#)

[stabsim2](#), [24](#)

[ttc](#), [3](#), [25](#)