Package matchingMarkets

December 14, 2013

`daa` *Deferred Acceptance Algorithm*

Description

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

Value

'daa' returns a list with the following items.

`s.prefs`	students' preference matrix.
`c.prefs`	colleges' preference matrix.
`iterations`	number of interations required to find the stable matching.
`matches`	identifier of students (men) assigned to colleges (women).
`match.mat`	matching matrix of dimension `nStudents x nColleges`.
`singles`	identifier of single/unmatched students (men).

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 <thilo@klein.co.uk>

References

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

Examples

```
## Marriage problem (3 men, 2 women) with random preferences:
daa(nStudents=3, nColleges=2)

## Marriage problem (3 men, 2 women) with 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) with random preferences:
daa(nStudents=7, nSlots=c(3,3))

## College admission problem (7 students, 2 colleges
## with 3 slots each) with 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))
```

plp *Partitioning Linear Programme*

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

`N`	number of players in the market
`V`	valuation matrix of dimension `NxN` that gives row-players valuation over column players (or vice versa)

Value

'plp' returns a list with the following items.

`Assignment.matrix`
: upper triangular matrix of dimension `NxN` 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 <thilo@klein.co.uk>

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)
```

smm *Structural Matching Model*

Description

Corrects for sample selection when the selection process is a one-sided matching game.

Usage

```
smm(selection, NTU, binary, offsetOut, offsetSel,
  marketFE, censored, dropOnes, repeatRun, niter, data,
  software)
```

Arguments

`selection`	logical: if TRUE estimate structural model with selection and outcome equation; if FALSE use outcome equation only.
`NTU`	logical: if TRUE use non-transferable utility (NTU) matching game; if FALSE use transferable utility (TU) matching game.
`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 use market-level fixed effects in outcome equation; if FALSE don't.
`censored`	delta is 0:not censored, 1:from below, 2:from above
`dropOnes`	one-group-markets exluded for estimation
`repeatRun`	repeated run
`niter`	number of iterations
`data`	data. see `Details` below
`software`	either `"matlab"` or `"octave"`

Value

'smm' returns a list with the following items.

`alphadraws`	matrix of dimension
`betadraws`	matrix of dimension
`deltadraws`	vector of length
`postmean`	vector of lenght
`poststd`	vector of lenght
`tstat`	vector of lenght

Author(s)

Thilo Klein <thilo@klein.co.uk>

References

Klein, T. (2014). matchingMarkets: An R Package for the Analysis of Stable Matchings.

Examples

```
############
## OCTAVE ##
############

## 1. Load RcppOctave
library(RcppOctave)

## 2. Run Gibbs sampler
mdata <- system.file("scripts/MatchingData.mat", package="matchingMarkets")
res <- smm(selection=1,NTU=1,binary=1,offsetOut=0,offsetSel=0,marketFE=0
  ,censored=0,dropOnes=0,repeatRun=0,niter=10,data=mdata,software="octave")

## 3. Get results
res$postmean
plot(res$alphadraws[1,], type="l")

############
## MATLAB ##
############

## 1. Load R.matlab
library(R.matlab)

## 2. Start MATLAB
Matlab$startServer()

## 3. Create a MATLAB client object used to communicate with MATLAB
matlab <- Matlab()
isOpen <- open(matlab) ## Connect to the MATLAB server

## 4. Run Gibbs sampler
mdata <- system.file("scripts/MatchingData.mat", package="matchingMarkets")
smm(selection=1,NTU=1,binary=1,offsetOut=0,offsetSel=0,marketFE=0,censored=0,
  dropOnes=0,repeatRun=0,niter=10,data=mdata,software="matlab")

## 5. Get results
res <- getVariable(matlab, c("alphadraws","betadraws","deltadraws",
  "postmean","poststd","tstat"))
res$postmean
plot(res$alphadraws[1,], type="l")

## 6. Close MATLAB server
close(matlab)
```

ttc *Top-Trading-Cycles Algorithm*

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 <thilo@klein.co.uk>

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