Package ‘KnapsackSampling’

October 16, 2016

Title Generate Feasible Samples of a Knapsack Problem

Version 0.1.0

Date 2016-10-15

Author Chin Soon Lim [aut]

Maintainer Chin Soon Lim <chinsoon12@hotmail.com>

Description The sampl.mcmc() function creates samples of the feasible region of a knapsack problem with both equalities and inequalities constraints.

Depends R (>= 3.3.0)

Imports lpSolve, utils, stats

License GPL (>= 2) | file LICENSE

RoxygenNote 5.0.1

URL https://github.com/chinsoon12/KnapsackSampling

BugReports https://github.com/chinsoon12/KnapsackSampling

NeedsCompilation no

Repository CRAN

Date/Publication 2016-10-16 15:13:33

R topics documented:

`flip01` *Flip a 1 and a 0 simultaneously*

Description

Flip a 1 and a 0 simultaneously

Usage

```
flip01(x)
```

Arguments

`x`	an integer or logical vector

Value

x an integer vector

`initState` *Generate an initial feasible solution by solving a linear programming with binary variables*

Description

Generate an initial feasible solution by solving a linear programming with binary variables

Usage

```
initState(numVar, objVec = runif(numVar), constraints = NULL)
```

Arguments

`numVar`	- number of variables
`objVec`	- objective function as a numeric vector
`constraints`	- a list of list of constraints with constr.mat, constr.dir, constr.rhs in each sublist

Value

a binary vector containing a feasible solution

Examples

```
#see documentation for sampl.mcmc
```

`sampl.mcmc` *Generate feasible solutions to a knapsack problem using Markov Chain Monte Carlo*

Description

Generate feasible solutions to a knapsack problem using Markov Chain Monte Carlo

Usage

```
sampl.mcmc(init, numSampl, maxIter = 2 * numSampl, constraints = NULL)
```

Arguments

`init`	- an initial feasible solution
`numSampl`	- number of samples to be generated
`maxIter`	- maximum number of iterations to be run to prevent infinite loop
`constraints`	- a list of list of constraints with constr.mat, constr.dir, constr.rhs in each sublist Please see example for an example of constraints.

Value

a matrix of 0, 1 with each row representing a sample

Examples

```
#number of variables
N <- 100

#number of variables in each group
grpLen <- 10

#equality matrix
A <- matrix(c(rep(1, N)), ncol=N, byrow=TRUE)

#inequality matrix
G <- matrix(c(rep(1, grpLen), rep(0, N - grpLen),
    rep(c(0,1), each=grpLen), rep(0, N - 2*grpLen)), ncol=N, byrow=TRUE)

#construct a list of list of constraints
constraints <- list(
    list(constr.mat=A, constr.dir=rep("==", nrow(A)), constr.rhs=c(20)),
    list(constr.mat=G, constr.dir=rep("<=", nrow(G)), constr.rhs=c(5, 5)),
    list(constr.mat=G, constr.dir=rep(">=", nrow(G)), constr.rhs=c(1, 2))
)

#generate an initial feasible solution
```

```
init <- initState(N, constraints=constraints)

#create feasible solutions to knapsack problems subject to constraints
samples <- sampl.mcmc(init, 50, constraints=constraints)
```

unlist.constr *Unpack constraints in a list of list into a matrix, equality or inequality signs, constant on right hand side*

Description

Unpack constraints in a list of list into a matrix, equality or inequality signs, constant on right hand side

Usage

```
unlist.constr(constraints)
```

Arguments

`constraints`	- a list of list of constraints with constr.mat, constr.dir, constr.rhs in each sublist

Value

a list containing matrix, signs and RHS constants

%[[% *Accessing the same named list elements of a list of lists*

Description

Accessing the same named list elements of a list of lists

Usage

```
x %[[% n
```

Arguments

`x`	- a list of sublists with same elements in each sublist
`n`	- name of element to be accessed from each sublist

Value

list of elements of name n in each sublist

References

credits to @kohske, Accessing same named list elements of the list of lists in R, http://stackoverflow.com/questions/5935673/a
same-named-list-elements-of-the-list-of-lists-in-r/5936077#5936077

Index