Package 'SDPDmod'

April 11, 2025

Title Spatial Dynamic Panel Data Modeling

Version 0.0.6

Description Spatial model calculation for static and dynamic panel data models, weights matrix creation and Bayesian model comparison.
Bayesian model comparison methods were described by 'LeSage' (2014) <doi:10.1016/j.spasta.2014.02.002>.
The 'Lee'-'Yu' transformation approach is described in 'Yu', 'De Jong' and 'Lee' (2008) <doi:10.1016/j.jeconom.2008.08.002>, 'Lee' and 'Yu' (2010) <doi:10.1016/j.jecon

License GPL (>= 3)

Depends R (>= 2.10)

Imports Matrix, methods, plm, RSpectra, sf, sp, spdep, stats

Suggests knitr, rmarkdown, splm

BugReports `https://github.com/RozetaSimonovska/SDPDmod/issues/`

VignetteBuilder knitr

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

NeedsCompilation no

Author Rozeta Simonovska [aut, cre]

Maintainer Rozeta Simonovska <`simonovska.r@gmail.com`>

Repository CRAN

Date/Publication 2025-04-11 19:40:02 UTC

Contents

blmpSDPD 2
coef.SDPDm 4
DDistMat 5
DistWMat 6
eignor 7

ExpDistMat 8
gN3dist 9
impactsSDPDm 9
InvDistMat 10
isrownor 11
mNearestN 12
mOrdNbr 13
print.blmpSDPD 14
print.SDPDm 14
print.summary.impactsSDPDm 15
print.summary.SDPDm 15
rownor 16
SDPDm 17
SharedBMat 19
summary.impactsSDPDm 20
summary.SDPDm 21
usa46 21

Index **22**

`blmpSDPD` *Bayesian log-marginal posterior probabilities for spatial panel models*

Description

Calculates log-marginal posterior probabilities for model comparison purposes.

Usage

```
blmpSDPD(
  formula,
  data,
  W,
  index,
  model = list("ols", "slx", "sar", "sdm", "sem", "sdem"),
  effect = "individual",
  ldet = NULL,
  lndetspec = list(m = NULL, p = NULL, sd = NULL),
  dynamic = FALSE,
  tlaginfo = list(ind = NULL),
  LYtrans = FALSE,
  incr = NULL,
  rintrv = TRUE,
  prior = "uniform",
  bprarg = 1.01
)
```

Arguments

`formula`	a symbolic description for the model to be estimated
`data`	a data.frame
`W`	spatial weights matrix (row-normalized)
`index`	the indexes (names of the variables for the spatial and time component)
`model`	a list of models for which the Bayesian log-marginal posterior probabilities need to be calculated, list("ols","slx","sar","sdm","sem","sdem")
`effect`	type of fixed effects, c("none","individual","time","twoways"), default ="individual"
`ldet`	Type of computation of log-determinant, c("full","mc"). Default "full" for smaller problems, "mc" for large problems.
`lndetspec`	specifications for the calculation of the log-determinant
`dynamic`	logical, if TRUE time lag of the dependent variable is included. Default = FALSE
`tlaginfo`	specification for the time lag, default = list(ind=NULL), *ind* - i-th column in the data frame which represents the time lag
`LYtrans`	logical, default FALSE. If Lee-Yu transformation should be used for demeaning of the variables
`incr`	increment for vector of values for rho
`rintrv`	logical, default TRUE, calculates eigenvalues of W. If FALSE, the interval for rho is (-1,1).
`prior`	type of prior to be used c("uniform","beta"). Default "uniform"
`bprarg`	argument for the beta prior. Default = 1.01

Details

For the Spatial Durbin Error Model (SDEM) the marginal distribution is:

$$p(\lambda|y) = \frac{1}{p(y)} p(\lambda)\Gamma(a)(2\pi)^{-a}\frac{|P|^{T-1}}{|Z'Z|^{1/2}}(e'e)^{-a}$$

For the Spatial Durbin Model (SDM) the marginal distribution is:

$$p(\rho|y) = \frac{1}{p(y)} p(\rho)\Gamma(a)(2\pi)^{-a}\frac{|P|}{|Z'Z|^{1/2}}(e'e)^{-a}$$

where $p(\lambda)$ is prior on λ and $p(\rho)$ is prior on ρ, either uniform $\frac{1}{D}$, $D = 1/\omega_{max} - 1/\omega_{min}$ or beta prior; No priors on beta and sige; ω_{max} and ω_{min} are the maximum and minimum eigenvalues of W - spatial weights matrix; $Z = X$ for lag or error model and $Z = [XWX]$ for Durbin model; X - matrix of k covariates.

For more details, see LeSage (2014).

Based on MatLab function log_marginal_panelprob.m.

In *tlaginfo = list(ind = NULL)*:

ind i-th column in *data* which represents the time lag, if not specified then the lag from the dependent variable is created and the panel is reduced from n*t to n*(t-1)

Value

A list

`lmarginal`	log-marginal posterior
`probs`	model probability

Author(s)

Rozeta Simonovska

References

LeSage, J. P., & Parent, O. (2007). Bayesian model averaging for spatial econometric models. *Geographical Analysis, 39(3)*, 241-267.

LeSage, J. P. (2014). Spatial econometric panel data model specification: A Bayesian approach. *Spatial Statistics, 9*, 122-145.

Examples

```
## US States Production data
data(Produc, package = "plm")
## Spatial weights row-normalized matrix of 48 US states
data(usaww, package = "splm")
isrownor(usaww)
form1 <- log(gsp) ~ log(pcap) + log(pc) + log(emp) + unemp
res1  <- blmpSDPD(formula = form1, data=Produc, W = usaww,
                 index = c("state","year"),
                 model = list("sar","sdm","sem","sdem"),
                 effect = "twoways")
res1
res2  <- blmpSDPD(formula = form1, data = Produc, W = usaww,
                 index = c("state","year"),
                 model = list("sar","sdm","sem","sdem"),
                 effect = "twoways", dynamic = TRUE)
res2
```

`coef.SDPDm` *Extract coefficients from model of class SDPDm*

Description

Method for extracting coefficients of objects of class "SDPDm"

Usage

```
## S3 method for class 'SDPDm'
coef(object, ...)
```

Arguments

`object`	object of class "SDPDm"
`...`	additional arguments to be passed

Value

Coefficients extracted from the model object of class "SDPDm".

Author(s)

Rozeta Simonovska

See Also

`SDPDm`

`DDistMat` *Double-Power Distance Weights Matrix*

Description

This function calculates the double-power distance matrix, for a given distance cutoff and a positive exponent.

Usage

```
DDistMat(distMat, distCutOff = NULL, powr = 2, mevn = FALSE)
```

Arguments

`distMat`	distance matrix
`distCutOff`	distance cutoff. Default = the maximal value from the distance matrix.
`powr`	power (positive exponent), default 2
`mevn`	logical, default FALSE. If TRUE, max-eigenvalue normalization is performed.

Details

W is an *nxn* matrix with elements $w_{ij}, i, j = 1, ...n$, where $w_{ij} = (1 - (\frac{d_{ij}}{D})^p)^p$, if $0 <= d_{ij} < D$ and $w_{ij} = 0$, if $d_{ij} > D$ or $i = j$. *D* is the cut-off distance point (maximum radius of influence), d_{ij} is the distance between spatial units *i* and *j*, and *p* is the power value (e.g. *p* = 2, 3, 4,...).

Value

`W`	spatial weights matrix (Default, not normalized)

Author(s)

Rozeta Simonovska

Examples

```
data(gN3dist) ##distance in meters
W1    <- DDistMat(distMat = gN3dist,
                  distCutOff = 300000,
                  powr = 3) ##distance cutoff in meters
dist2 <- gN3dist/1000 ##in km
W2    <- DDistMat(distMat = dist2, 300, 3)  ##distance cutoff in kilometers
```

DistWMat *Distance weights matrix (Inverse distance, Exponential distance or Double-Distance matrix)*

Description

This function calculates the spatial distance weights matrix (inverse, exponential or double-distance), with a given cutoff distance and a positive exponent (alpha).

Usage

```
DistWMat(
  distMat,
  distCutOff = NULL,
  type = "inverse",
  alpha = NULL,
  mevn = FALSE
)
```

Arguments

`distMat`	distance matrix
`distCutOff`	cutoff distance. Default = the maximal value from the distance matrix.
`type`	the type of distance matrix c("inverse","expo","doubled"). Default = "inverse".
`alpha`	power (positive exponent), default 1 if type="inverse", 0.01 if type="expo" and 2 if type="double"
`mevn`	logical, default FALSE. If TRUE, max-eigenvalue normalization is performed.

Value

`W`	spatial weights matrix (Default, not normalized)

Author(s)

Rozeta Simonovska

See Also

InvDistMat ExpDistMat DDistMat vignette("spatial_matrices", package = "SDPDmod")

Examples

```
## distance between centroids of NUTS3 regions in Germany (in meters)
data(gN3dist, package = "SDPDmod")
##inverse distance matrix with cutoff 100000 meters
W1    <- DistWMat(distMat = gN3dist, distCutOff = 100000)
dist2 <- gN3dist/1000 ##distance in km
## normalized exponential distance matrix
W2    <- DistWMat(distMat=dist2, distCutOff = 100, type = "expo",
                  alpha = 2, mevn = TRUE)
```

eignor *Maximum eigenvalue normalization*

Description

Maximum eigenvalue row normalization of a spatial weights matrix.

Usage

```
eignor(W)
```

Arguments

W	spatial weights matrix

Value

W	Eigenvalue normalized spatial weights matrix

Author(s)

Rozeta Simonovska

See Also

rownor

Examples

```
data(gN3dist)
dist2 <- gN3dist/1000 ##distance in km
W     <- InvDistMat(distMat = dist2, distCutOff = 100, powr = 2)
Wnor  <- eignor(W)
```

`ExpDistMat` *Exponential distance matrix*

Description

This function calculates the (negative) exponential distance matrix, with a given cutoff distance and a positive exponent value.

Usage

```
ExpDistMat(distMat, distCutOff = NULL, expn = 0.01, mevn = FALSE)
```

Arguments

`distMat`	distance matrix
`distCutOff`	cutoff distance. Default = the maximal value from the distance matrix.
`expn`	positive exponent, default = 0.01
`mevn`	logical, default FALSE. If TRUE, max-eigenvalue normalization is performed.

Details

W is an *nxn* matrix with elements w_{ij}, *i, j = 1,..n*, where $w_{ij} = e^{-\alpha d_{ij}}$, if $0 <= d_{ij} < D$ and $w_{ij} = 0$, if $d_{ij} > D$ or $i = j$. *D* is the distance cutoff point (maximum radius of influence), d_{ij} is the distance between spatial units *i* and *j*, and α is the positive exponent (e.g. α= 0.01, 0.02,...).

Value

`W`	spatial weights matrix (Default, not normalized)

Author(s)

Rozeta Simonovska

Examples

```
data(gN3dist) ##distance in meters
W1    <- ExpDistMat(distMat = gN3dist, distCutOff = 100000)
dist2 <- gN3dist/1000 ##in km
W2    <- ExpDistMat(distMat = dist2, distCutOff = 100, expn = 0.02)
W2nor <- ExpDistMat(distMat = dist2, 100000, 0.001, mevn = TRUE)
```

`gN3dist` *Distance between the centroids of NUTS3 regions in Germany*

Description

Distance between the centroids of NUTS3 regions in Germany

Usage

```
gN3dist
```

Format

matrix of distances

`impactsSDPDm` *Impacts for 'SDPDm' objects*

Description

Direct and indirect effects estimates

Usage

```
impactsSDPDm(res, NSIM = 200, sd = 12345)
```

Arguments

`res`	an object of class 'SDPDm'
`NSIM`	number of simulations to be performed, default = 200
`sd`	starting seed, default = 12345

Details

For spatial dynamic panel data model:

$$y_t = \tau y_{t-1} + \rho W y_t + \eta W y_{t-1} + X_t \beta + W X_t \theta + \alpha + \mu + u_t$$

Short term effects for k*th* explanatory variable:

$$(I - \rho W)^{-1} (\beta_k I_n + \theta_k W)$$

Long term effects for k*th* explanatory variable:

$$((1 - \tau) I_n - (\rho + \eta) W)^{-1} (\beta_k I_n + \theta_k W)$$

The direct effect is the average of the diagonal elements, and the indirect effect is the average of the row sums of the non-diagonal elements of the matrix.

Value

An object of class 'impactsSDPDm'

Author(s)

Rozeta Simonovska

See Also

`SDPDm`

`InvDistMat` *Inverse distance matrix*

Description

This function calculates the inverse distances, with a given cutoff distance and a positive exponent.

Usage

```
InvDistMat(distMat, distCutOff = NULL, powr = 1, mevn = FALSE)
```

Arguments

`distMat`	distance matrix
`distCutOff`	cutoff distance. Default = the maximal value from the distance matrix.
`powr`	power (positive exponent), default = 1
`mevn`	logical, default FALSE. If TRUE, max-eigenvalue normalization is performed.

Details

W is an *nxn* matrix with elements w_{ij}, *i,j=1,..n*, where $w_{ij} = 1/d_{ij}^{\gamma}$, if 0 $<= d_{ij} < D$ and $w_{ij} = 0$, if $d_{ij} > D$ or $i = j$. *D* is the distance cutoff point (maximum radius of influence), d_{ij} is the distance between spatial units *i* and *j*, and γ is the value for the exponent (e.g. γ = 1, 2, 3, 4,...).

Value

`W`	weights matrix (Default, not normalized)

Author(s)

Rozeta Simonovska

Examples

```
## distance between centroids of NUTS3 regions in Germany (in meters)
data(gN3dist, package = "SDPDmod")
## inverse distance matrix with cutoff 100000 meters
W1    <- InvDistMat(distMat = gN3dist, distCutOff = 100000)
dist2 <- gN3dist/1000 ##distance in km
## normalized distance matrix with cutoff 100km
W2    <- InvDistMat(distMat = dist2, distCutOff=100, powr = 2, mevn = TRUE)
```

`isrownor` *Is the matrix row-normalized*

Description

Checks if a spatial weights matrix is row-normalized.

Usage

```
isrownor(W)
```

Arguments

`W`	spatial weights matrix

Value

Logical value. If the weights matrix is row-normalized such that all rows sum up to 1, the value is TRUE.

Author(s)

Rozeta Simonovska

See Also

`rownor`

Examples

```
data("usa46", package="SDPDmod")
isrownor(usa46)
```

mNearestN *m nearest neighbors based on a distance matrix*

Description

This function finds the m nearest neighbors, given a matrix of distances.

Usage

```
mNearestN(distMat, m = 5, listv = FALSE, rn = FALSE)
```

Arguments

`distMat`	distance matrix
`m`	number of nearest neighbors, default value 5
`listv`	logical, default FALSE. If TRUE the list of neighbors should also be returned
`rn`	logical, default FALSE. If TRUE, the spatial weights matrix will be row-normalized

Value

`W`	spatial weights matrix
`nlist`	list of indexes of the m nearest neighbors

Author(s)

Rozeta Simonovska

Examples

```
data(gN3dist, package = "SDPDmod")
fournn <- mNearestN(gN3dist, m = 4)
mat1   <- rownor(fournn)
tennn  <- mNearestN(gN3dist, 10, listv = TRUE, rn = TRUE)
mat2   <- tennn$W
```

mOrdNbr *1st to m-th order neighbors matrix*

Description

Finds the 1th to m-th order neighbors matrix.

Usage

```
mOrdNbr(sf_pol = NULL, m = 1, neigbs = NULL, listv = FALSE, rn = FALSE)
```

Arguments

`sf_pol`	spatial polygons object
`m`	the order of neighbors up to which they will be included in the weights matrix, default 1
`neigbs`	neighbors list, default NULL
`listv`	logical, default FALSE. If TRUE the list of neighbors should also be returned
`rn`	logical, default FALSE. If TRUE, the weight matrix will be row-normalized

Value

`W`	spatial weights matrix
`nlist`	list of neighbors

Author(s)

Rozeta Simonovska

Examples

```
library("sf")
ger   <- st_read(system.file("shape/GermanyNUTS3.shp",
                             package = "SDPDmod"),
                 quiet = TRUE)
m1thn <- mOrdNbr(ger)

m4thn <- mOrdNbr(ger, 4)
mat1  <- rownor(m4thn)
m4thn2<- mOrdNbr(ger, 4, listv = TRUE, rn = TRUE)
mat2  <- m4thn2$W
```

print.blmpSDPD *Print for class blmpSDPD*

Description

Method for printing the results of objects of class "blmpSDPD"

Usage

```
## S3 method for class 'blmpSDPD'
print(x, digits = max(3, getOption("digits") - 3), ...)
```

Arguments

`x`	object of class "blmpSDPD"
`digits`	number of digits
`...`	additional arguments to be passed

Value

No return value

Author(s)

Rozeta Simonovska

print.SDPDm *print for class SDPDm*

Description

Method for sprinting the results of objects of class "SDPDm"

Usage

```
## S3 method for class 'SDPDm'
print(x, digits = max(3, getOption("digits") - 3), ...)
```

Arguments

`x`	object of class "SDPDm"
`digits`	number of digits
`...`	additional arguments to be passed

Value

No return value

Author(s)

Rozeta Simonovska

See Also

`SDPDm`

`print.summary.impactsSDPDm` *Print summary for class impactsSDPDm*

Description

Method for printing the summary the results of objects of class "impactsSDPDm"

Usage

```
## S3 method for class 'summary.impactsSDPDm'
print(x, ...)
```

Arguments

`x`	summary object of class "impactsSDPDm"
`...`	additional arguments to be passed

Author(s)

Rozeta Simonovska

`print.summary.SDPDm` *Print of summary for class SDPDm*

Description

Method for printing the summary the results of objects of class "SDPDm"

Usage

```
## S3 method for class 'summary.SDPDm'
print(x, ...)
```

Arguments

`x`	summary object of class "SDPDm"
`...`	additional arguments to be passed

Value

No return value

Author(s)

Rozeta Simonovska

See Also

`SDPDm`

`rownor` *Row-normalization*

Description

Row-normalization of a spatial weights matrix.

Usage

```
rownor(W)
```

Arguments

`W`	spatial weights matrix

Value

`W`	row-normalized spatial weights matrix

Author(s)

Rozeta Simonovska

See Also

`eignor`

Examples

```
library("sf")
ger   <- st_read(system.file("shape/GermanyNUTS3.shp",
                             package = "SDPDmod"),
                 quiet = TRUE)
W     <- mOrdNbr(ger, 3)
Wnor  <- rownor(W)
```

`SDPDm` *Spatial dynamic panel data lag model with fixed effects maximum likelihood estimation.*

Description

This function estimates spatial panel model with fixed effects for static or dynamic model. It includes the transformation approach suggested by Yu et al (2008) and Lee and Yu (2010).

Usage

```
SDPDm(
  formula,
  data,
  W,
  index,
  model = "sar",
  effect = "individual",
  ldet = NULL,
  lndetspec = list(p = NULL, m = NULL, sd = NULL),
  dynamic = FALSE,
  tlaginfo = list(ind = NULL, tl = TRUE, stl = TRUE),
  LYtrans = TRUE,
  incr = NULL,
  rintrv = TRUE,
  demn = FALSE,
  DIRtrans = FALSE
)
```

Arguments

`formula`	a symbolic description for the (static) model to be estimated, not including the dynamic component
`data`	a data.frame
`W`	spatial weights matrix
`index`	the indexes (Names of the variables for the spatial and time component. The spatial is first and the time second.)

`model`	a models to be calculated, c("sar","sdm"), default = "sar"
`effect`	type of fixed effects, c("none","individual","time","twoways"), default ="individual"
`ldet`	type of computation of log-determinant, c("full","mc"). Default "full" for smaller problems, "mc" for large problems.
`lndetspec`	specifications for the calculation of the log-determinant for mcmc calculation. Default list(p=NULL,m=NULL,sd=NULL), if the number of spatial units is >1000 then list(p=30,m=30,sd=12345)
`dynamic`	logical, if TRUE time lag of the dependent variable is included. Default = FALSE
`tlaginfo`	specification for the time lag, default = list(ind = NULL, tl = TRUE, stl = TRUE), see details
`LYtrans`	logical, default TRUE. If the Lee-Yu transformation should be used for bias correction
`incr`	increment for vector of values for rho
`rintrv`	logical, default TRUE, calculates eigenvalues of W. If FALSE, the interval for rho is (-1,1)
`demn`	logical, if Lee-Yu transformation for demeaning of the variables to remove fixed effects is performed (only used in static models). Default FALSE
`DIRtrans`	logical, if direct transformation of variables should be used. Default, FALSE (only used in dynamic models with "twoways" effects)

Details

Based on MatLab functions sar_jihai.m, sar_jihai_time.m and sar_panel_FE.m

In *tlaginfo = list(ind = NULL, tl = TRUE, stl = TRUE)*:

ind i-th column in *data* which represents the time lag, if not specified then the lag from the dependent variable is created and the panel is reduced from n*t to n*(t-1)

tl logical, default TRUE. If TRUE y_{t-1} (the lagged dependent variable in time is included)

stl logical, default TRUE. If TRUE Wy_{t-1} (the lagged dependent variable in space and time is included)

Value

An object of class "SDPDm"

`coefficients`	coefficients estimate of the model parameters (*coefficients1* for dynamic model)
`rho`	spatial coefficient
`sige`	residuals variance
`llik`	the value of the log likelihood function
`...`	

Author(s)

Rozeta Simonovska

References

Yu, J., De Jong, R., & Lee, L. F. (2008). Quasi-maximum likelihood estimators for spatial dynamic panel data with fixed effects when both n and T are large. *Journal of Econometrics*, 146(1), 118-134.

Lee, L. F., & Yu, J. (2010). Estimation of spatial autoregressive panel data models with fixed effects. *Journal of Econometrics*, 154(2), 165-185.

Lee, L. F., & Yu, J. (2010). A spatial dynamic panel data model with both time and individual fixed effects. *Econometric Theory*, 564-597.

See Also

`vignette("spatial_model", package = "SDPDmod")`

Examples

```
library("SDPDmod")
data(Produc, package = "plm")
data(usaww, package = "splm")
form1 <- log(gsp) ~ log(pcap) + log(pc) + log(emp) + unemp
mod1  <- SDPDm(formula = form1, data = Produc, W = usaww, index = c("state","year"),
               model = "sar", effect = "individual", LYtrans = TRUE)
summary(mod1)
imp1  <- impactsSDPDm(mod1)
summary(imp1)
mod2  <- SDPDm(formula = form1, data = Produc, W = usaww, index = c("state","year"),
               model = "sdm", effect = "twoways", LYtrans = TRUE,
               dynamic = TRUE, tlaginfo=list(ind = NULL, tl = TRUE, stl = TRUE))
summary(mod2)
```

SharedBMat *Shared boundary matrix*

Description

This function calculates the shared boundary matrix

Usage

```
SharedBMat(sf_pol, rn = FALSE)
```

Arguments

sf_pol	spatial polygons, spatial lines object or spatial data frame
rn	logical, default FALSE. If TRUE, the spatial weights matrix is row-normalized

Value

`W`	spatial weights matrix (length of shared boundary between spatial units)

Author(s)

Rozeta Simonovska

Examples

```
library("sf")

ger   <- st_read(system.file("shape/GermanyNUTS3.shp",
                             package = "SDPDmod"),
                 quiet = TRUE)
bav <- ger[which(substr(ger$NUTS_CODE,1,3)=="DE2"),] ## Bavarian districts
W     <- SharedBMat(bav)
```

`summary.impactsSDPDm` *Summary for class impactsSDPDm*

Description

Method for summarizing the results of objects of class "impactsSDPDm"

Usage

```
## S3 method for class 'impactsSDPDm'
summary(object, ...)
```

Arguments

`object`	object of class "impactsSDPDm"
`...`	additional arguments to be passed

Value

Summary of impacts

Author(s)

Rozeta Simonovska

See Also

`SDPDm`

`summary.SDPDm` *Summary for class SDPDm*

Description

Method for summarizing the results of objects of class "SDPDm"

Usage

```
## S3 method for class 'SDPDm'
summary(object, ...)
```

Arguments

`object`	object of class "SDPDm"
`...`	additional arguments to be passed

Value

Summary of SDPDm

Author(s)

Rozeta Simonovska

See Also

`SDPDm`

`usa46` *Spatial weights matrix of 46 USA states*

Description

Spatial weights matrix of 46 USA states

Usage

```
usa46
```

Format

binary coded matrix

Index

∗ **data**
    gN3dist, 9
    usa46, 21

blmpSDPD, 2

coef.SDPDm, 4

DDistMat, 5, *7*
DistWMat, 6

eignor, 7, *16*
ExpDistMat, *7*, 8

gN3dist, 9

impactsSDPDm, 9
InvDistMat, *7*, 10
isrownor, 11

mNearestN, 12
mOrdNbr, 13

print.blmpSDPD, 14
print.SDPDm, 14
print.summary.impactsSDPDm, 15
print.summary.SDPDm, 15

rownor, *7*, *11*, 16

SDPDm, *10*, 17
SharedBMat, 19
summary.impactsSDPDm, 20
summary.SDPDm, 21

usa46, 21