Package ‘SDPDmod’

December 12, 2022

Title Spatial Dynamic Panel Data Modeling

Version 0.0.1

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>.


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 7.2.1

NeedsCompilation no

Author Rozeta Simonovska [aut, cre]

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

Repository CRAN

Date/Publication 2022-12-12 11:50:02 UTC

R topics documented:

blmpSDPD .............................................. 2
DDistMat ............................................. 4
DistWMat ............................................ 5
eignor ................................................. 6
ExpDistMat .......................................... 7
Bayesian log-marginal posterior probabilities for spatial panel models

Description
Calculates log-marginal posterior probabilities for model comparison purposes.

Usage
```r
blmpSDPD(
  formula,
  data,
  W,
  index,
  model,
  effect,
  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 ="none"

ldet  
Type of computation of log-determinant, c("full","mc"). Default "full" for smaller problems, "mc" for large problems.

ldetspec  
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

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

where \(p(\rho)\) is prior on \(\rho\), either uniform \(\frac{1}{D}\), \(D = 1/\omega_{\text{max}} - 1/\omega_{\text{min}}\) or beta prior; No priors on beta and sige; \(\omega_{\text{max}}\) and \(\omega_{\text{min}}\) are the maximum and minimum eigenvalues of \(W\) - spatial weights matrix;

\(a = (NT - 2k)/2\), \(k\) - number of covariates;

\(|P| = |I_N - \rho W|\); 

\(Z = X\) for lag or error model and \(Z = [XW X]\) for Durbin model;

\(e = y - Z \delta; \delta = |Z'Z|^{-1}Z'y\)

Based on MatLab function log_marginal_panelprob.m

Value

A list

lmarginal  
log-marginal posterior

probs  
model probability

Author(s)

Rozeta Simonovska
References


Examples

```r
data(Produc, package = "plm") ## US States Production data
data(usaww, package = "splm") ## Spatial weights row-normalized matrix of 48 US states
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
```

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

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

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>distMat</td>
<td>distance matrix</td>
</tr>
<tr>
<td>distCutOff</td>
<td>distance cutoff. Default = the maximal value from the distance matrix.</td>
</tr>
<tr>
<td>powr</td>
<td>power (positive exponent), default 2</td>
</tr>
<tr>
<td>mevn</td>
<td>logical, default FALSE. If TRUE, max-eigenvalue normalization is performed.</td>
</tr>
</tbody>
</table>

Details

$W$ is an $n \times n$ matrix with elements $w_{ij}, i, j = 1, \ldots, n$, where $w_{ij} = (1 - \left(\frac{d_{ij}}{D}\right)^p)^p$, if $0 \leq d_{ij} < D$ 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. $p = 2, 3, 4, \ldots$).
DistWMat

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
ExpDistMat

Examples

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

ExpDistMat  

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 $mn$ 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 $\alpha$ is the positive exponent (e.g. $\alpha = 0.01, 0.02...$).

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)
Distance between the centroids of NUTS3 regions in Germany

Usage
gN3dist

Format
matrix of distances

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 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:

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

**Value**

An object of class `impactsSDPDm`

**Author(s)**

Rozeta Simonovska

**See Also**

SDPDm

---

<table>
<thead>
<tr>
<th>InvDistMat</th>
<th>Inverse distance matrix</th>
</tr>
</thead>
</table>

**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 $n \times n$ matrix with elements $w_{ij}$, $i,j=1...n$, where $w_{ij} = 1/d_{ij}^\gamma$, if $0 \leq d_{ij} < D$ 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 $\gamma$ is the value for the exponent (e.g. $\gamma = 1, 2, 3, 4,...$).

**Value**

$W$ weights matrix (Default, not normalized)

**Author(s)**

Rozeta Simonovska
Examples

```r
## 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

```r
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`
Description

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

Usage

```r
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

```r
data(gN3dist, package = "SDPDmod")
fournn <- mNearestN(gN3dist, m = 4)
mat1 <- rrownor(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

```r
library("sf")
ger <- st_read(system.file(dsn = "shape/GermanyNUTS3.shp",package = "SDPDmod"))
m1thn <- mOrdNbr(ger)
m4thn <- mOrdNbr(ger, 4)
mat1 <- rownor(m4thn)
m4thn2<- mOrdNbr(ger, 4, listv = TRUE, rn = TRUE)
mat2 <- m4thn2$W
```
**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**

```r
library("sf")
ger <- st_read(system.file(dsn = "shape/GermanyNUTS3.shp", package = "SDPDmod"))
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, effect, ldet = NULL, lndetspec = list(p = NULL, m = NULL, sd = NULL),
  dynamic = FALSE, tlaginfo = list(ind = NULL, tl = FALSE, stl = FALSE), LYtrans = FALSE, 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 ="none"
ldet type of computation of log-determinant, c("full","mc"). Default "full" for smaller problems, "mc" for large problems.
lnetspec 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=FALSE,stl=FALSE), see details
LYtrans logical, default FALSE. 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(p = NULL, m = NULL, sd = NULL): ind i-th column in data which represents the time lag tl logical, default FALSE. If TRUE yt−1 (the lagged dependent variable in time is included) stl logical, default FALSE. If TRUE Wy−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


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)
SharedBMat <- SDPDm(formula = form1, data = Produc, W = usaww, index = c("state", "year"),
                     model = "sdm", effect = "twoways", LYtrans = TRUE,
                     dynamic = TRUE, tiaginfo = 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**

```r
library("sf")

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

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

**Usage**

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

**Arguments**

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

**Value**

No return value

**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, 8
  usa46, 17
blmpSDPD, 2

DDistMat, 4, 6
DistWMat, 5
eignor, 6, 13
ExpDistMat, 6, 7
gN3dist, 8
impactsSDPDm, 8
InvDistMat, 6, 9
isrownor, 10

mNearestN, 11
mOrdNbr, 12

rownor, 6, 10, 13
SDPDm, 9, 13
SharedBMat, 16
summary.SDPDm, 17
usa46, 17