Package ‘estimateW’

December 6, 2022

Type Package

Title Estimation of Spatial Weight Matrices

Version 0.0.1

Author Tamas Krisztin [aut, cre] (<https://orcid.org/0000-0002-9241-8628>),
Philipp Piribauer [aut] (<https://orcid.org/0000-0002-3414-3101>)

Maintainer Tamas Krisztin <krisztin@iiasa.ac.at>

Description

Bayesian estimation of spatial weight matrices in spatial econometric panel models. Allows for estimation of spatial autoregressive (SAR), spatial Durbin (SDM), and spatially lagged explanatory variable (SLX) type specifications featuring an unknown spatial weight matrix. Methodological details are given in Krisztin and Piribauer (2022) <doi:10.1080/17421772.2022.2095426>.

License GPL (>= 3)

Depends R (>= 3.5.0)

Encoding UTF-8

Language EN-US

LazyData true

Imports Matrix, matrixcalc, plot.matrix, stats, R6

RoxygenNote 7.2.2

NeedsCompilation no

Repository CRAN

Date/Publication 2022-12-06 13:30:02 UTC

R topics documented:

`bbinompdf` *Probability density for a hierarchical prior setup for the elements of the adjacency matrix based on the beta binomial distribution*

Description

A hierarchical prior setup can be used in `W_priors` to anchor the prior number of expected neighbors. Assuming a *fixed* prior inclusion probability $\underline{p} = 1/2$ for the off-diagonal entries in the binary n by n adjacency matrix Ω implies that the number of neighbors (i.e. the row sums of Ω) follows a Binomial distribution with a prior expected number of neighbors for the n spatial observations of $(n-1)\underline{p}$. However, such a prior structure has the potential undesirable effect of promoting a relatively large number of neighbors. To put more prior weight on parsimonious neighborhood structures and promote sparsity in Ω, the beta binomial prior accounts for the number of neighbors in each row of Ω.

Usage

```
bbinompdf(x, nsize, a, b, min_k = 0, max_k = nsize)
```

Arguments

`x`	Number of neighbors (scalar)
`nsize`	Number of potential neighbors: `nsize`$= (n-1)$
`a`	Scalar prior parameter a
`b`	Scalar prior parameter b
`min_k`	Minimum prior number of neighbors (defaults to 0)
`max_k`	Maximum prior number of neighbors (defaults to `nsize`)

Details

The beta-binomial distribution is the result of treating the prior inclusion probability $\underline{p}$ as random (rather than being fixed) by placing a hierarchical beta prior on it. For the number of neighbors x, the resulting prior on the elements of Ω, ω_{ij}, can be written as:

$$p(\omega_{ij} = 1|x) \propto \Gamma\left(a + x\right)\Gamma\left(b + (n-1) - x\right),$$

where $\Gamma(\cdot)$ is the Gamma function, and a and b are hyperparameters from the beta prior. In the case of $a = b = 1$, the prior takes the form of a discrete uniform distribution over the number of neighbors. By fixing $a = 1$ the prior can be anchored around the expected number of neighbors m through $b = [(n-1) - m]/m$ (see Ley and Steel, 2009).

The prior can be truncated by setting a minimum (`min_k`) and/or a maximum number of neighbors (`max_k`). Values outside this range have zero prior support.

Value

Prior density evaluated at `x`.

References

Ley, E., & Steel, M. F. (2009). On the effect of prior assumptions in Bayesian model averaging with applications to growth regression. *Journal of Applied Econometrics*, **24(4)**. doi:10.1002/jae.1057.

`betapdf` *The four-parameter Beta probability density function*

Description

A four-parameter Beta specification as the prior for the spatial autoregressive parameter ρ, as proposed by LeSage and Parent (2007) .

Usage

```
betapdf(rho, a = 1, b = 1, rmin = 0, rmax = 1)
```

Arguments

`rho`	The scalar value for ρ
`a`	The first shape parameter of the Beta distribution
`b`	The second shape parameter of the Beta distribution
`rmin`	Scalar $\underline{\rho}_{min}$: the minimum value of ρ
`rmax`	Scalar $\underline{\rho}_{max}$: the maximum value of ρ

Details

The prior density is given by:

$$p(\rho) \sim \frac{1}{Beta(a,b)} \frac{(\rho - \underline{\rho}_{min})^{(a-1)}(\underline{\rho}_{max} - \rho)^{(b-1)}}{2^{a+b-1}}$$

where $Beta(a,b)$ $(a, b > 0)$ represents the Beta function, $Beta(a,b) = \int_0^1 t^{a-1}(1-t)^{b-1}dt$.

Value

Density value evaluated at `rho`.

References

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

`beta_priors` *Set prior specifications for the slope parameters*

Description

This function allows the user to specify custom values for Gaussian priors on the slope parameters.

Usage

```
beta_priors(
  k,
  beta_mean_prior = matrix(0, k, 1),
  beta_var_prior = diag(k) * 100
)
```

Arguments

`k`	The total number of slope parameters in the model.
`beta_mean_prior`	numeric k by 1 matrix of prior means $\underline{\mu}_{\beta}$.
`beta_var_prior`	A k by k matrix of prior variances $\underline{V}_{\beta}$. Defaults to a diagonal matrix with `100` on the main diagonal.

Details

For the slope parameters β the package uses common Normal prior specifications. Specifically, $p(\beta) \sim \mathcal{N}(\underline{\mu}_{\beta}, \underline{V}_{\beta})$.

This function allows the user to specify custom values for the prior hyperparameters $\underline{\mu}_{\beta}$ and $\underline{V}_{\beta}$. The default values correspond to weakly informative Gaussian priors with mean zero and a diagonal prior variance-covariance matrix with 100 on the main diagonal.

Value

A list with the prior mean vector (`beta_mean_prior`), the prior variance matrix (`beta_var_prior`) and the inverse of the prior variance matrix (`beta_var_prior_inv`).

`beta_sampler` *An R6 class for sampling slope parameters*

Description

This class samples slope parameters with a Gaussian prior from the conditional posterior. Use the beta_priors class for setup.

Format

An `R6Class` generator object

Public fields

`beta_prior` The current `beta_priors`

`curr_beta` The current value of β

Methods

Public methods:

- `beta_sampler$new()`
- `beta_sampler$sample()`

Method `new()`:

Usage:

```
beta_sampler$new(beta_prior)
```

Arguments:

`beta_prior` The list returned by `beta_priors`

Method `sample()`:

Usage:

```
beta_sampler$sample(Y, X, curr_sigma)
```

Arguments:

`Y` The N by 1 matrix of responses

`X` The N by k design matrix

`curr_sigma` The variance parameter σ^2

covid *Covid incidences data*

Description

COVID-19 data set provided by Johns Hopkins University (Dong et al., 2020). The database contains information on (official) daily infections for a large panel of countries around the globe in the very beginning of the outbreak from 17 February to 20 April 2020.

Usage

```
covid
```

Format

A `data.frame` object.

Details

Data is provided for countries: Australia (AUS), Bahrain (BHR), Belgium (BEL), Canada (CAN), China (CHN), Finland (FIN), France (FRA), Germany (DEU), Iran (IRN), Iraq (IRQ), Israel (ISR), Italy (ITA), Japan (JPN), Kuwait (KWT), Lebanon (LBN), Malaysia (MYS), Oman (OMN), Republic of Korea (KOR), Russian Federation (RUS), Singapore (SGP), Spain (ESP), Sweden (SWE), Thailand (THA), United Arab Emirates (ARE), United Kingdom (GBR), United States of America (USA), and Viet Nam (VNM).

The dataset includes daily data on the country specific maximum measured temperature (Temperature) and precipitation levels (Precipitation) as additional covariates (source: Dark Sky API). The stringency index (Stringency) put forward by Hale et al. (2020), which summarizes country-specific governmental policy measures to contain the spread of the virus. We use the biweekly average of the reported stringency index.

References

Dong, E., Du, H., and Gardner, L. (2020). An interactive web-based dashboard to track COVID-19 in real time. *The Lancet Infectious Diseases*, **20(5)**, 533–534. doi:10.1016/S14733099(20)301201.

Hale, T., Petherick, A., Phillips, T., and Webster, S. (2020). Variation in government responses to COVID-19. Blavatnik School of Government Working Paper, 31, 2020–2011. doi:10.1038/s41562-021010798.

Krisztin, T., and Piribauer, P. (2022). A Bayesian approach for the estimation of weight matrices in spatial autoregressive models, *Spatial Economic Analysis*, 1-20. doi:10.1080/17421772.2022.2095426.

Krisztin, T., Piribauer, P., and Wögerer, M. (2020). The spatial econometrics of the coronavirus pandemic. *Letters in Spatial and Resource Sciences*, **13 (3)**, 209-218. doi:10.1007/s12076020-002541.

Dong, E., Du, H., and Gardner, L. (2020). An interactive web-based dashboard to track COVID-19 in real time. *The Lancet Infectious Diseases*, **20(5)**, 533–534. doi:10.1016/S14733099(20)301201.

logdetAinvUpdate — *Efficient update of the log-determinant and the matrix inverse*

Description

While updating the elements of the spatial weight matrix in SAR and SDM type spatial models in a MCMC sampler, the log-determinant term has to be regularly updated, too. When the binary elements of the adjacency matrix are treated unknown, the Matrix Determinant Lemma and the Sherman-Morrison formula are used for computationally efficient updates.

Usage

```
logdetAinvUpdate(ch_ind, diff, AI, logdet)
```

Arguments

`ch_ind`	vector of non-negative integers, between 1 and n. Denotes which rows of A should be updated.
`diff`	a numeric `length(ch_ind)` by `n` matrix. This value will be added to the corresponding rows of A.
`AI`	numeric n by n matrix that is the inverse of $A = (I_n - \rho W)$. This inverse will be updated using the Sherman-Morrison formula.
`logdet`	single number that is the log-determinant of the matrix A. This log-determinant will be updated through the Matrix Determinant Lemma.

Details

Let $A = (I_n - \rho W)$ be an invertible n by n matrix. v is an n by 1 column vector of real numbers and u is a binary vector containing a single one and zeros otherwise. Then the Matrix Determinant Lemma states that:

$$A + uv' = (1 + v'A^{-1}u)det(A)$$

.

This provides an update to the determinant, but the inverse of A has to be updated as well. The Sherman-Morrison formula proves useful:

$$(A + uv')^{-1} = A^{-1}\frac{A^{-1}uv'A^{-1}}{1 + v'A^{-1}u}$$

.

Using these two formulas, an efficient update of the spatial projection matrix determinant can be achieved.

Value

A list containing the updated n by n matrix A^{-1}, as well as the updated log determinant of A

References

Sherman, J., and Morrison, W. J. (1950) Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *The Annals of Mathematical Statistics*, **21(1)**, 124-127.

Harville, D. A. (1998) Matrix algebra from a statistician's perspective. Taylor & Francis.

`logdetPaceBarry` *Pace and Barry's log determinant approximation*

Description

Bayesian estimates of parameters of SAR and SDM type spatial models require the computation of the log-determinant of positive-definite spatial projection matrices of the form $(I_n - \rho W)$, where W is a n by n spatial weight matrix. However, direct computation of the log-determinant is computationally expensive.

Usage

```
logdetPaceBarry(W, length.out = 200, rmin = -1, rmax = 1)
```

Arguments

`W`	numeric n by n non-negative spatial weights matrix, with zeros on the main diagonal.
`length.out`	single, integer number, has to be at least 51 (due to order of approximation). Sets how fine the grid approximation is. Default value is 200.
`rmin`	single number between -1 and 1. Sets the minimum value of the spatial autoregressive parameter ρ. Has to be lower than `rmax`. Default value is -1.
`rmax`	single number between -1 and 1. Sets the maximum value of the spatial autoregressive parameter ρ. Has to be higher than `rmin`. Default value is 1.

Details

This function wraps the log-determinant approximation by Barry and Pace (1999), which can be used to precompute the log-determinants over a grid of ρ values.

Value

numeric `length.out` by `2` matrix; the first column contains the approximated log-determinants the second column the ρ values ranging between `rmin` and `rmax`.

References

Barry, R. P., and Pace, R. K. (1999) Monte Carlo estimates of the log determinant of large sparse matrices. *Linear Algebra and its applications*, **289(1-3)**, 41-54.

`normalgamma` *A Markov Chain Monte Carlo (MCMC) sampler for a linear panel model*

Description

The sampler uses independent Normal-inverse-Gamma priors to estimate a linear panel data model. The function is used for an illustration on using the `beta_sampler` and `sigma_sampler` classes.

Usage

```
normalgamma(
  Y,
  tt,
  X = matrix(1, nrow(Y), 1),
  niter = 200,
  nretain = 100,
  beta_prior = beta_priors(k = ncol(X)),
  sigma_prior = sigma_priors()
)
```

Arguments

`Y`	numeric $N \times 1$ matrix containing the dependent variables, where $N = nT$ is the number of spatial (n) times the number of time observations (T, with `tt=T`). Note that the observations have organized such that $Y = [Y_1', ..., Y_T']'$.
`tt`	single number greater or equal to 1. Denotes the number of time observations. $tt = T$.
`X`	numeric $N \times k_1$ design matrix of independent variables.
`niter`	single number greater or equal to 1, indicating the total number of draws. Will be automatically coerced to integer. The default value is 200.
`nretain`	single number greater or equal to 0, indicating the number of draws kept after the burn-in. Will be automatically coerced to integer. The default value is 100.
`beta_prior`	list containing priors for the slope coefficients β, generated by the smart constructor `beta_priors`.
`sigma_prior`	list containing priors for the error variance σ^2, generated by the smart constructor `sigma_priors`

Details

The considered model takes the form:

$$Y_t = X_t\beta + \varepsilon_t,$$

with $\varepsilon_t \sim N(0, I_n\sigma^2)$.

Y_t $(n \times 1)$ collects the n cross-sectional observations for time $t = 1, ..., T$. X_t $(n \times k_1)$ is a matrix of explanatory variables. β $(k_1 \times 1)$ is an unknown slope parameter matrix.

After vertically staking the T cross-sections $Y = [Y_1', ..., Y_T']'$ $(N \times 1)$, $X = [X_1', ..., X_T']'$ $(N \times k)$, with $N = nT$, the final model can be expressed as:

$$Y = X\beta + \varepsilon,$$

where $\varepsilon \sim N(0, I_N \sigma^2)$. Note that the input data matrices have to be ordered first by the cross-sectional (spatial) units and then stacked by time.

Examples

```
n = 20; tt = 10; k = 3
X = matrix(stats::rnorm(n*tt*k),n*tt,k)
Y = X %*% c(1,0,-1) + stats::rnorm(n*tt,0,.5)
res = normalgamma(Y,tt,X)
```

`plot.estimateW` *Graphical summary of the estimated adjacency matrix* Ω

Description

Graphical plot of the posterior probabilities of the estimated adjacency matrix Ω.

Usage

```
## S3 method for class 'estimateW'
plot(
  x,
  cols = c("white", "lightgrey", "black"),
  breaks = c(0, 0.5, 0.75, 1),
  ...
)
```

Arguments

`x`	`estimateW` object.
`cols`	Main colors to use for the plot
`breaks`	Breaks for the colors
`...`	further arguments are passed on to be invoked

plot.sim_dgp — *Graphical summary of a generated spatial weight matrix*

Description

Graphical summary of a generated spatial weight matrix

Usage

```
## S3 method for class 'sim_dgp'
plot(x, ...)
```

Arguments

x	sim_dgp object
...	further arguments are passed on to the invoked

rho_priors — *Specify prior for the spatial autoregressive parameter and sampling settings*

Description

Specify prior for the spatial autoregressive parameter and sampling settings

Usage

```
rho_priors(
  rho_a_prior = 1,
  rho_b_prior = 1,
  rho_min = 0,
  rho_max = 1,
  init_rho_scale = 1,
  griddy_n = 60,
  use_griddy_gibbs = TRUE,
  mh_tune_low = 0.4,
  mh_tune_high = 0.6,
  mh_tune_scale = 0.1
)
```

Arguments

- `rho_a_prior` — Single number. Prior hyperparameter for the four-parameter beta distribution `betapdf`. Defaults to 1.
- `rho_b_prior` — Single number. Prior hyperparameter for the four-parameter beta distribution `betapdf`. Defaults to 1.
- `rho_min` — Minimum value for ρ (default: 0)
- `rho_max` — Maximum value for ρ (default: 1)
- `init_rho_scale` — For Metropolis-Hastings step the initial candidate variance (default: 1)
- `griddy_n` — single integer number. Sets how fine the grid approximation is. Default value is 60.
- `use_griddy_gibbs` — Binary value. Should griddy-Gibbs be used for ρ estimation? `use_griddy_gibbs=TRUE` does not work if `row_standardized_prior = FALSE` is specified in the W prior specification. if TRUE: griddy-Gibbs step for sampling ρ; if FALSE: tuned random-walk Metropolis-Hastings step
- `mh_tune_low` — Lower bound of acceptance rate for Metropolis-Hastings tuning (used if `use_griddy_gibbs==FALSE`)
- `mh_tune_high` — Upper bound of acceptance rate for Metropolis-Hastings tuning (used if `use_griddy_gibbs==FALSE`)
- `mh_tune_scale` — Scaling factor for Metropolis-Hastings tuning (used if `use_griddy_gibbs==FALSE`)

---

rho_sampler — *An R6 class for sampling the spatial autoregressive parameter ρ*

Description

An R6 class for sampling the spatial autoregressive parameter ρ

An R6 class for sampling the spatial autoregressive parameter ρ

Format

An `R6Class` generator object

Details

This class samples the spatial autoregressive parameter using either a tuned random-walk Metropolis-Hastings or a griddy Gibbs step. Use the `rho_priors` class for setup.

For the Griddy-Gibbs algorithm see Ritter and Tanner (1992).

Public fields

rho_prior The current rho_priors

curr_rho The current value of ρ

curr_W The current spatial weight matrix W; an n by n matrix.

curr_A The current spatial filter matrix $I - \rho W$.

curr_AI The inverse of curr_A

curr_logdet The current log-determinant of curr_A

curr_logdets A set of log-determinants for various values of ρ. See the rho_priors function for settings of step site and other parameters of the grid.

Methods

Public methods:

- rho_sampler$new()
- rho_sampler$stopMHtune()
- rho_sampler$setW()
- rho_sampler$sample()
- rho_sampler$sample_Griddy()
- rho_sampler$sample_MH()

Method new()**:**

Usage:

```
rho_sampler$new(rho_prior, W = NULL)
```

Arguments:

rho_prior The list returned by rho_priors

W An optional starting value for the spatial weight matrix W

Method stopMHtune()**:** Function to stop the tuning of the Metropolis-Hastings step. The tuning of the Metropolis-Hastings step is usually carried out until half of the burn-in phase. Call this function to turn it off.

Usage:

```
rho_sampler$stopMHtune()
```

Method setW()**:**

Usage:

```
rho_sampler$setW(newW, newLogdet = NULL, newA = NULL, newAI = NULL)
```

Arguments:

newW The updated spatial weight matrix W.

newLogdet An optional value for the log determinant corresponding to newW and curr_rho.

newA An optional value for the spatial projection matrix using newW and curr_rho.

newAI An optional value for the matrix inverse of newA.

Method sample()**:**

Usage:

```
rho_sampler$sample(Y, mu, sigma)
```

Arguments:

`Y` The n by T matrix of responses.

`mu` The n by T matrix of means.

`sigma` The variance parameter σ^2.

Method `sample_Griddy()`**:**

Usage:

```
rho_sampler$sample_Griddy(Y, mu, sigma)
```

Arguments:

`Y` The n by T matrix of responses.

`mu` The n by T matrix of means.

`sigma` The variance parameter σ^2.

Method `sample_MH()`**:**

Usage:

```
rho_sampler$sample_MH(Y, mu, sigma)
```

Arguments:

`Y` The n by T matrix of responses.

`mu` The n by T matrix of means.

`sigma` The variance parameter σ^2.

References

Ritter, C., and Tanner, M. A. (1992). Facilitating the Gibbs sampler: The Gibbs stopper and the griddy-Gibbs sampler. *Journal of the American Statistical Association*, **87(419)**, 861-868.

`sar` *A Markov Chain Monte Carlo (MCMC) sampler for the panel spatial autoregressive model (SAR) with exogenous spatial weight matrix.*

Description

The sampler uses independent Normal-inverse-Gamma priors for the slope and variance parameters, as well as a four-parameter beta prior for the spatial autoregressive parameter ρ. The function is used as an illustration on using the `beta_sampler`, `sigma_sampler`, and `rho_sampler` classes.

Usage

```
sar(
  Y,
  tt,
  W,
  Z = matrix(1, nrow(Y), 1),
  niter = 200,
  nretain = 100,
  rho_prior = rho_priors(),
  beta_prior = beta_priors(k = ncol(Z)),
  sigma_prior = sigma_priors()
)
```

Arguments

`Y`	numeric $N \times 1$ matrix containing the dependent variables, where $N = nT$ is the number of spatial (n) times the number of time observations (T, with `tt=T`). Note that the observations have organized such that $Y = [Y_1', ..., Y_T']'$.
`tt`	single number greater or equal to 1. Denotes the number of time observations. $tt = T$.
`W`	numeric, non-negative and row-stochastic n by n exogenous spatial weight matrix. Must have zeros on the main diagonal.
`Z`	numeric $N \times k_3$ design matrix of independent variables. The default value is a $N \times 1$ vector of ones (i.e. an intercept for the model).
`niter`	single number greater or equal to 1, indicating the total number of draws. Will be automatically coerced to integer. The default value is 200.
`nretain`	single number greater or equal to 0, indicating the number of draws kept after the burn-in. Will be automatically coerced to integer. The default value is 100.
`rho_prior`	list of prior settings for estimating ρ, generated by the smart constructor `rho_priors`
`beta_prior`	list containing priors for the slope coefficients, generated by the smart constructor `beta_priors`.
`sigma_prior`	list containing priors for the error variance σ^2, generated by the smart constructor `sigma_priors`

Details

The considered panel spatial autoregressive model (SAR) takes the form:

$$Y_t = \rho W Y_t + Z_t \beta + \varepsilon_t,$$

with $\varepsilon_t \sim N(0, I_n \sigma^2)$. The row-stochastic n by n spatial weight matrix W is non-negative and has zeros on the main diagonal. ρ is a scalar spatial autoregressive parameter.

Y_t $(n \times 1)$ collects the n cross-sectional (spatial) observations for time $t = 1, ..., T$. Z_t $(n \times k_3)$ is a matrix of explanatory variables. β $(k_3 \times 1)$ is an unknown slope parameter matrix.

After vertically staking the T cross-sections $Y = [Y_1', ..., Y_T']'$ $(N \times 1)$, $Z = [Z_1', ..., Z_T']'$ $(N \times k_3)$, with $N = nT$, the final model can be expressed as:

$$Y = \rho \tilde{W} Y + Z\beta + \varepsilon,$$

where $\tilde{W} = I_T \otimes W$ and $\varepsilon \sim N(0, I_N \sigma^2)$. Note that the input data matrices have to be ordered first by the cross-sectional spatial units and then stacked by time. This is a wrapper function calling sdm with no spatially lagged dependent variables.

Examples

```
n = 20; tt = 10
dgp_dat = sim_dgp(n =n, tt = tt, rho = .5, beta3 = c(1,.5), sigma2 = .5)
res = sar(Y = dgp_dat$Y,tt = tt, W = dgp_dat$W,
         Z = dgp_dat$Z,niter = 100,nretain = 50)
```

---

sarw — *A Markov Chain Monte Carlo (MCMC) sampler for the panel spatial autoregressive model (SAR) with unknown spatial weight matrix*

Description

The sampler uses independent Normal-inverse-Gamma priors for the slope and variance parameters, as well as a four-parameter beta prior for the spatial autoregressive parameter ρ. This is a wrapper function calling sdmw with no spatially lagged exogenous variables.

Usage

```
sarw(
  Y,
  tt,
  Z,
  niter = 100,
  nretain = 50,
  W_prior = W_priors(n = nrow(Y)/tt),
  rho_prior = rho_priors(),
  beta_prior = beta_priors(k = ncol(Z)),
  sigma_prior = sigma_priors()
)
```

Arguments

Y	numeric $N \times 1$ matrix containing the dependent variables, where $N = nT$ is the number of spatial (n) times the number of time observations (T, with tt=T). Note that the observations have organized such that $Y = [Y_1', ..., Y_T']'$.
tt	single number greater or equal to 1. Denotes the number of time observations. $tt = T$.
Z	numeric $N \times k_3$ design matrix of independent variables. The default value is a $N \times 1$ vector of ones (i.e. an intercept for the model).

niter	single number greater or equal to 1, indicating the total number of draws. Will be automatically coerced to integer. The default value is 100.
nretain	single number greater or equal to 0, indicating the number of draws kept after the burn-in. Will be automatically coerced to integer. The default value is 50.
W_prior	list containing prior settings for estimating the spatial weight matrix W. Generated by the smart constructor W_priors.
rho_prior	list of prior settings for estimating ρ, generated by the smart constructor rho_priors
beta_prior	list containing priors for the slope coefficients β, generated by the smart constructor beta_priors.
sigma_prior	list containing priors for the error variance σ^2, generated by the smart constructor sigma_priors

Details

The considered panel spatial autoregressive model (SAR) with unknown (n by n) spatial weight matrix W takes the form:

$$Y_t = \rho W Y_t + Z\beta + \varepsilon_t,$$

with $\varepsilon_t \sim N(0, I_n\sigma^2)$ and $W = f(\Omega)$. The n by n matrix Ω is an unknown binary adjacency matrix with zeros on the main diagonal and $f(\cdot)$ is the (optional) row-standardization function. ρ is a scalar spatial autoregressive parameter.

Y_t ($n \times 1$) collects the n cross-sectional (spatial) observations for time $t = 1, ..., T$. Z_t ($n \times k_3$) is a matrix of explanatory variables. β ($k_3 \times 1$) is an unknown slope parameter vector.

After vertically staking the T cross-sections $Y = [Y_1', ..., Y_T']'$ ($N \times 1$), and $Z = [Z_1', ..., Z_T']'$ ($N \times k_3$), with $N = nT$. The final model can be expressed as:

$$Y = \rho\tilde{W}Y + Z\beta + \varepsilon,$$

where $\tilde{W} = I_T \otimes W$ and $\varepsilon \sim N(0, I_N\sigma^2)$. Note that the input data matrices have to be ordered first by the cross-sectional spatial units and then stacked by time.

Estimation usually even works well in cases of $n >> T$. However, note that for applications with $n > 200$ the estimation process becomes computationally demanding and slow. Consider in this case reducing niter and nretain and carefully check whether the posterior chains have converged.

Value

List with posterior samples for the slope parameters, ρ, σ^2, W, and average direct, indirect, and total effects.

Examples

```
n = 20; tt = 10
dgp_dat = sim_dgp(n =n, tt = tt, rho = .5, beta3 = c(.5,1),
            sigma2 = .05,n_neighbor = 3,intercept = TRUE)
res = sarw(Y = dgp_dat$Y,tt = tt,Z = dgp_dat$Z,niter = 20,nretain = 10)
```

sdm — *A Markov Chain Monte Carlo (MCMC) sampler for the panel spatial Durbin model (SDM) with exogenous spatial weight matrix.*

Description

The sampler uses independent Normal-inverse-Gamma priors for the slope and variance parameters, as well as a four-parameter prior for the spatial autoregressive parameter ρ. The function is used as an illustration on using the `beta_sampler`, `sigma_sampler`, and `rho_sampler` classes.

Usage

```
sdm(
  Y,
  tt,
  W,
  X = matrix(0, nrow(Y), 0),
  Z = matrix(1, nrow(Y), 1),
  niter = 200,
  nretain = 100,
  rho_prior = rho_priors(),
  beta_prior = beta_priors(k = ncol(X) * 2 + ncol(Z)),
  sigma_prior = sigma_priors()
)
```

Arguments

`Y`	numeric $N \times 1$ matrix containing the dependent variables, where $N = nT$ is the number of spatial (n) times the number of time observations (T, with `tt=T`). Note that the observations have organized such that $Y = [Y_1', ..., Y_T']'$.
`tt`	single number greater or equal to 1. Denotes the number of time observations. $tt = T$.
`W`	numeric, non-negative and row-stochastic n by n exogenous spatial weight matrix. Must have zeros on the main diagonal.
`X`	numeric $N \times k_1$ design matrix of independent variables. These will be automatically spatially lagged. If no spatially lagged variable is included in the model a matrix with N rows and zero columns should be supplied (the default value). Note: either X or Z has to be a matrix with at least one column.
`Z`	numeric $N \times k_3$ design matrix of independent variables which are not spatially lagged. The default value is a $N \times 1$ vector of ones (i.e. an intercept for the model). Note: either X or Z has to be a matrix with at least one column.
`niter`	single number greater or equal to 1, indicating the total number of draws. Will be automatically coerced to integer. The default value is 200.
`nretain`	single number greater or equal to 0, indicating the number of draws kept after the burn-in. Will be automatically coerced to integer. The default value is 100.

`rho_prior`	list of prior settings for estimating ρ, generated by the smart constructor `rho_priors`
`beta_prior`	list containing priors for the slope coefficients β, generated by the smart constructor `beta_priors`. The ordering of the priors is: (1) priors of X, (2) priors of spatially lagged X, (3) priors of Z.
`sigma_prior`	list containing priors for the error variance σ^2, generated by the smart constructor `sigma_priors`

Details

The considered panel spatial Durbin model (SDM) takes the form:

$$Y_t = \rho W Y_t + X_t \beta_1 + W X_t \beta_2 + Z \beta_3 + \varepsilon_t,$$

with $\varepsilon_t \sim N(0, I_n \sigma^2)$. The row-stochastic n by n spatial weight matrix W is non-negative and has zeros on the main diagonal. ρ is a scalar spatial autoregressive parameter.

Y_t $(n \times 1)$ collects the n cross-sectional (spatial) observations for time $t = 1, ..., T$. X_t $(n \times k_1)$ and Z_t $(n \times k_2)$ are matrices of explanatory variables, where the former will also be spatially lagged. β_1 $(k_1 \times 1)$, β_2 $(k_1 \times 1)$ and β_3 $(k_2 \times 1)$ are unknown slope parameter vectors.

After vertically staking the T cross-sections $Y = [Y_1', ..., Y_T']'$ $(N \times 1)$, $X = [X_1', ..., X_T']'$ $(N \times k_1)$ and $Z = [Z_1', ..., Z_T']'$ $(N \times k_2)$, with $N = nT$, the final model can be expressed as:

$$Y = \rho \tilde{W} Y + X \beta_1 + \tilde{W} X \beta_2 + Z \beta_3 + \varepsilon,$$

where $\tilde{W} = I_T \otimes W$ and $\varepsilon \sim N(0, I_N \sigma^2)$. Note that the input data matrices have to be ordered first by the cross-sectional spatial units and then stacked by time.

Examples

```
n = 20; tt = 10
dgp_dat = sim_dgp(n = n, tt = tt, rho = .5, beta1 = c(.5,1), beta2 = c(-1,.5),
                  beta3 = c(1.5), sigma2 = .5)
res = sdm(Y = dgp_dat$Y, tt = tt,  W = dgp_dat$W, X = dgp_dat$X,
          Z = dgp_dat$Z, niter = 100, nretain = 50)
```

`sdmw` *A Markov Chain Monte Carlo (MCMC) sampler for the panel spatial Durbin model (SDM) with unknown spatial weight matrix*

Description

The sampler uses independent Normal-inverse-Gamma priors for the slope and variance parameters, as well as a four-parameter beta prior for the spatial autoregressive parameter ρ. It is a wrapper around `W_sampler`.

Usage

```
sdmw(
  Y,
  tt,
  X = matrix(0, nrow(Y), 0),
  Z = matrix(1, nrow(Y), 1),
  niter = 100,
  nretain = 50,
  W_prior = W_priors(n = nrow(Y)/tt),
  rho_prior = rho_priors(),
  beta_prior = beta_priors(k = ncol(X) * 2 + ncol(Z)),
  sigma_prior = sigma_priors()
)
```

Arguments

`Y`	numeric $N \times 1$ matrix containing the dependent variables, where $N = nT$ is the number of spatial (n) times the number of time observations (T, with `tt=T`). Note that the observations have organized such that $Y = [Y_1', ..., Y_T']'$.
`tt`	single number greater or equal to 1. Denotes the number of time observations. $tt = T$.
`X`	numeric $N \times k_1$ design matrix of independent variables. These will be automatically spatially lagged. If no spatially lagged variable is included in the model a matrix with N rows and zero columns should be supplied (the default value). Note: either X or Z has to be a matrix with at least one column.
`Z`	numeric $N \times k_3$ design matrix of independent variables which are not spatially lagged. The default value is a $N \times 1$ vector of ones (i.e. an intercept for the model). Note: either X or Z has to be a matrix with at least one column.
`niter`	single number greater or equal to 1, indicating the total number of draws. Will be automatically coerced to integer. The default value is 100.
`nretain`	single number greater or equal to 0, indicating the number of draws kept after the burn-in. Will be automatically coerced to integer. The default value is 50.
`W_prior`	list containing prior settings for estimating the spatial weight matrix W. Generated by the smart constructor `W_priors`.
`rho_prior`	list of prior settings for estimating ρ, generated by the smart constructor `rho_priors`
`beta_prior`	list containing priors for the slope coefficients β, generated by the smart constructor `beta_priors`. The ordering of the priors is: (1) priors of X, (2) priors of spatially lagged X, (3) priors of Z.
`sigma_prior`	list containing priors for the error variance σ^2, generated by the smart constructor `sigma_priors`

Details

The considered panel spatial Durbin model (SDM) with unknown (n by n) spatial weight matrix W takes the form:

$$Y_t = \rho W Y_t + X_t\beta_1 + W X_t\beta_2 + Z\beta_3 + \varepsilon_t,$$

with $\varepsilon_t \sim N(0, I_n\sigma^2)$ and $W = f(\Omega)$. The n by n matrix Ω is an unknown binary adjacency matrix with zeros on the main diagonal and $f(\cdot)$ is the (optional) row-standardization function. ρ is a scalar spatial autoregressive parameter.

Y_t $(n \times 1)$ collects the n cross-sectional (spatial) observations for time $t = 1, ..., T$. X_t $(n \times k_1)$ and Z_t $(n \times k_2)$ are matrices of explanatory variables, where the former will also be spatially lagged. β_1 $(k_1 \times 1)$, β_2 $(k_1 \times 1)$ and β_3 $(k_2 \times 1)$ are unknown slope parameter vectors.

After vertically staking the T cross-sections $Y = [Y_1', ..., Y_T']'$ $(N \times 1)$, $X = [X_1', ..., X_T']'$ $(N \times k_1)$ and $Z = [Z_1', ..., Z_T']'$ $(N \times k_2)$, with $N = nT$. The final model can be expressed as:

$$Y = \rho\tilde{W}Y + X\beta_1 + \tilde{W}X\beta_2 + Z\beta_3 + \varepsilon,$$

where $\tilde{W} = I_T \otimes W$ and $\varepsilon \sim N(0, I_N\sigma^2)$. Note that the input data matrices have to be ordered first by the cross-sectional spatial units and then stacked by time.

Estimation usually even works well in cases of $n >> T$. However, note that for applications with $n > 200$ the estimation process becomes computationally demanding and slow. Consider in this case reducing `niter` and `nretain` and carefully check whether the posterior chains have converged.

Value

List with posterior samples for the slope parameters, ρ, σ^2, W, and average direct, indirect, and total effects.

Examples

```
n = 20; tt = 10
dgp_dat = sim_dgp(n =n, tt = tt, rho = .75, beta1 = c(.5,1),beta2 = c(-1,.5),
            beta3 = c(1.5), sigma2 = .05,n_neighbor = 3,intercept = TRUE)
res = sdmw(Y = dgp_dat$Y,tt = tt,X = dgp_dat$X,Z = dgp_dat$Z,niter = 20,nretain = 10)
```

---

sigma_priors — *Set prior specification for the error variance using an inverse Gamma distribution*

Description

Set prior specification for the error variance using an inverse Gamma distribution

Usage

```
sigma_priors(sigma_rate_prior = 0.001, sigma_shape_prior = 0.001)
```

Arguments

`sigma_rate_prior`
: Sigma rate prior parameter (scalar), default: 0.001.

`sigma_shape_prior`
: Sigma shape prior parameter (scalar), default: 0.001.
: This function allows the user to specify priors for the error variance σ^2.

`sigma_sampler` *An R6 class for sampling for sampling σˆ2*

Description

This class samples nuisance parameter which an inverse Gamma prior from the conditional posterior. Use the sigma_priors class for setup.

Format

An `R6Class` generator object

Public fields

`sigma_prior` The current `sigma_priors`

`curr_sigma` The current value of σ^2

Methods

Public methods:

- `sigma_sampler$new()`
- `sigma_sampler$sample()`

Method `new()`**:**

Usage:

```
sigma_sampler$new(sigma_prior)
```

Arguments:

`sigma_prior` The list returned by `sigma_priors`

Method `sample()`**:**

Usage:

```
sigma_sampler$sample(Y, mu)
```

Arguments:

`Y` The N by 1 matrix of responses

`mu` The N by 1 matrix of means

sim_dgp *Simulating from a data generating process*

Description

This function can be used to generate data from a data generating process for SDM, SAR, SLX type models.

Usage

```
sim_dgp(
  n,
  tt,
  rho,
  beta1 = c(),
  beta2 = c(),
  beta3 = c(),
  sigma2,
  n_neighbor = 4,
  do_symmetric = FALSE,
  intercept = FALSE
)
```

Arguments

n	Number of spatial observations n.
tt	Number of time observations T.
rho	The true ρ parameter
beta1	Vector of dimensions $k_1 \times 1$. Provides the values for β_1 Defaults to `c()`. Note: has to be of same length as β_2.
beta2	Vector of dimensions $k_1 \times 1$. Provides the values for β_2 Defaults to `c()`. Note: has to be fo same length as β_1.
beta3	Vector of dimensions $k_2 \times 1$. Provides the values for β_3 Defaults to `c()`.
sigma2	The true σ^2 parameter for the DGP. Has to be a scalar larger than zero.
n_neighbor	Number of neighbors for the generated $n \times n$ spatial weight W matrix. Defaults to 4.
do_symmetric	Should the generated spatial weight matrix be symmetric? (default: FALSE)
intercept	Should the first column of Z be an intercept? Defaults to `FALSE`. If `intercept = TRUE`, β_3 has to be at least of length `1`.

Details

The generated spatial panel model takes the form

$$Y = \rho W Y + X\beta_1 + W X\beta_2 + Z\beta_3 + \epsilon,$$

with $\epsilon \sim N(0, I_n\sigma^2)$. he function generates the $N \times 1$ vector Y. The elements of the explanatory variable matrices X ($N \times k_1$) and Z ($N \times k_2$) are randomly generated from a Gaussian distribution with zero mean and unity variance ($N(0, 1)$).

The non-negative, row-stochastic n by n matrix W is constructed using a k-nearest neighbor specification based on a randomly generated spatial location pattern, with coordinates sampled from a standard normal distribution.

Values for the parameters β_1, β_2, and β_3, as well as ρ and σ^2 have to be provided by the user. The length of β_1 and β_2 have to be equal.

- A spatial Durbin model (SDM) is constructed if ρ is not equal to zero and β_1, β_2, and β_3 are all supplied by the user.
- A spatial autoregressive model is constructed if ρ is not equal to zero and only β_3 is supplied by the user.
- An SLX type model is constructed if ρ is equal to zero and β_1, β_2 are supplied by the user.

Value

A list with the generated X, Y and W and a list of parameters.

Examples

```
# SDM data generating process
dgp_dat = sim_dgp(n =20, tt = 10, rho = .5, beta1 = c(1,-1),
                  beta2 = c(0,.5),beta3 = c(.2),sigma2 = .5)
```

---

slxw *A Markov Chain Monte Carlo (MCMC) sampler for the panel spatial SLX model with unknown spatial weight matrix*

Description

The sampler uses independent Normal-inverse-Gamma priors for the slope and variance parameters. It is a wrapper around `W_sampler`.

Usage

```
slxw(
  Y,
  tt,
  X = matrix(0, nrow(Y), 0),
  Z = matrix(1, nrow(Y), 1),
  niter = 100,
  nretain = 50,
  W_prior = W_priors(n = nrow(Y)/tt),
  beta_prior = beta_priors(k = ncol(X) * 2 + ncol(Z)),
  sigma_prior = sigma_priors()
)
```

Arguments

`Y`	numeric $N \times 1$ matrix containing the dependent variables, where $N = nT$ is the number of spatial (n) times the number of time observations (T, with `tt=T`). Note that the observations have organized such that $Y = [Y_1', ..., Y_T']'$.
`tt`	single number greater or equal to 1. Denotes the number of time observations. $tt = T$.
`X`	numeric $N \times k_1$ design matrix of independent variables. These will be automatically spatially lagged. If no spatially lagged variable is included in the model a matrix with N rows and zero columns should be supplied (the default value). Note: either X or Z has to be a matrix with at least one column.
`Z`	numeric $N \times k_3$ design matrix of independent variables which are not spatially lagged. The default value is a $N \times 1$ vector of ones (i.e. an intercept for the model). Note: either X or Z has to be a matrix with at least one column.
`niter`	single number greater or equal to 1, indicating the total number of draws. Will be automatically coerced to integer. The default value is 100.
`nretain`	single number greater or equal to 0, indicating the number of draws kept after the burn-in. Will be automatically coerced to integer. The default value is 50.
`W_prior`	list containing prior settings for estimating the spatial weight matrix W. Generated by the smart constructor `W_priors`.
`beta_prior`	list containing priors for the slope coefficients β, generated by the smart constructor `beta_priors`. The ordering of the priors is: (1) priors of X, (2) priors of spatially lagged X, (3) priors of Z.
`sigma_prior`	list containing priors for the error variance σ^2, generated by the smart constructor `sigma_priors`

Details

The considered spatial panel SLX model with unknown (n by n) spatial weight matrix W takes the form:

$$Y_t = X_t\beta_1 + WX_t\beta_2 + Z\beta_3 + \varepsilon_t,$$

with $\varepsilon_t \sim N(0, I_n\sigma^2)$ and $W = f(\Omega)$. The n by n matrix Ω is an unknown binary adjacency matrix with zeros on the main diagonal and $f(\cdot)$ is the (optional) row-standardization function.

Y_t $(n \times 1)$ collects the n cross-sectional (spatial) observations for time $t = 1, ..., T$. X_t $(n \times k_1)$ and Z_t $(n \times k_2)$ are matrices of explanatory variables, where the former will also be spatially lagged. β_1 $(k_1 \times 1)$, β_2 $(k_1 \times 1)$ and β_3 $(k_2 \times 1)$ are unknown slope parameter vectors.

After vertically staking the T cross-sections $Y = [Y_1', ..., Y_T']'$ $(N \times 1)$, $X = [X_1', ..., X_T']'$ $(N \times k_1)$ and $Z = [Z_1', ..., Z_T']'$ $(N \times k_2)$, with $N = nT$. The final model can be expressed as:

$$Y = X\beta_1 + \tilde{W}X\beta_2 + Z\beta_3 + \varepsilon,$$

where $\tilde{W} = I_T \otimes W$ and $\varepsilon \sim N(0, I_N\sigma^2)$. Note that the input data matrices have to be ordered first by the cross-sectional spatial units and then stacked by time.

Estimation usually even works well in cases of $n >> T$. However, note that for applications with $n > 200$ the estimation process becomes computationally demanding and slow. Consider in this case reducing `niter` and `nretain` and carefully check whether the posterior chains have converged.

Value

List with posterior samples for the slope parameters, σ^2, W, and average direct, indirect, and total effects.

Examples

```
set.seed(123)
n = 20; tt = 10
dgp_dat = sim_dgp(n = 20, tt = 10, rho = 0, beta1 = c(1,-1),
                  beta2 = c(3,-2.5), beta3 = c(.2), sigma2 = .05,
                  n_neighbor = 3,intercept = TRUE)
res = slxw(Y = dgp_dat$Y, tt = tt, X = dgp_dat$X, Z = dgp_dat$Z,
                  niter = 20, nretain = 10)
```

`W_priors` *Set prior specifications for the spatial weight matrix*

Description

Set prior specifications for the n by n spatial weight matrix $W = f(\Omega)$, where Ω is an n by n unknown binary adjacency matrix (with zeros on the main diagonal), and $f()$ denotes the (optional) row-standardization function

Usage

```
W_priors(
  n,
  W_prior = matrix(0.5, n, n),
  symmetric_prior = FALSE,
  row_standardized_prior = TRUE,
```

```
    nr_neighbors_prior = bbinompdf(0:(n - 1), nsize = n - 1, a = 1, b = 1, min_k = 0, max_k
      = n - 1)
)
```

Arguments

n
: The number of spatial observations

W_prior
: An n by n matrix of prior inclusion probabilities for W

symmetric_prior
: Binary value. Should the estimated adjacency matrix Ω be symmetric (default: FALSE)? if TRUE: Ω is forced symmetric; if FALSE: Ω not necessarily symmetric.

row_standardized_prior
: Binary value. Should the estimated W matrix be row-standardized (default: TRUE)? if TRUE: row-stochastic W; if FALSE: W not row-standardized.

nr_neighbors_prior
: An n dimensional vector of prior weights on the number of neighbors (i.e. the row sums of the adjacency matrix Ω), where the first element denotes the prior probability of zero neighbors and the last those of $n-1$. A prior using only fixed inclusion probabilities for the entries in Ω would be an n dimensional vector of $1/n$. Defaults to a bbinompdf prior, with prior parameters $a = 1$, $b = 1$.

W_sampler *An R6 class for sampling the elements of W*

Description

An R6 class for sampling the elements of W

An R6 class for sampling the elements of W

Format

An R6Class generator object

Details

This class samples the spatial weight matrix. Use the function W_priors class for setup.

The sampling procedure relies on conditional Bernoulli posteriors outlined in Krisztin and Piribauer (2022).

Public fields

W_prior The current W_priors

curr_w numeric, non-negative n by n spatial weight matrix with zeros on the main diagonal. Depending on the W_priors settings can be symmetric and/or row-standardized.

curr_W binary n by n spatial connectivity matrix Ω

`curr_A` The current spatial projection matrix $I - \rho W$.

`curr_AI` The inverse of `curr_A`

`curr_logdet` The current log-determinant of `curr_A`

`curr_rho` single number between -1 and 1 or NULL, depending on whether the sampler updates the spatial autoregressive parameter ρ. Set while invoking `initialize` or using the function `set_rho`.

Methods

Public methods:

- `W_sampler$new()`
- `W_sampler$set_rho()`
- `W_sampler$sample()`

Method `new()`:

Usage:

```
W_sampler$new(W_prior, curr_rho = NULL)
```

Arguments:

`W_prior` The list returned by `W_priors`

`curr_rho` optional single number between -1 and 1. Value of the spatial autoregressive parameter ρ. Defaults to NULL, in which case no updates of the log-determinant, the spatial projection matrix, and its inverse are carried out.

Method `set_rho()`: If the spatial autoregressive parameter ρ is updated during the sampling procedure the log determinant, the spatial projection matrix $I - \rho W$ and it's inverse must be updated. This function should be used for a consistent update. At least the new scalar value for ρ must be supplied.

Usage:

```
W_sampler$set_rho(new_rho, newLogdet = NULL, newA = NULL, newAI = NULL)
```

Arguments:

`new_rho` single, number; must be between -1 and 1.

`newLogdet` An optional value for the log determinant corresponding to `newW` and `curr_rho`

`newA` An optional value for the spatial projection matrix using `newW` and `curr_rho`

`newAI` An optional value for the matrix inverse of `newA`

Method `sample()`:

Usage:

```
W_sampler$sample(Y, curr_sigma, mu, lag_mu = matrix(0, nrow(tY), ncol(tY)))
```

Arguments:

`Y` The n by tt matrix of responses

`curr_sigma` The variance parameter σ^2

`mu` The n by tt matrix of means.

`lag_mu` n by tt matrix of means that will be spatially lagged with the estimated W. Defaults to a matrix with zero elements.

References

Krisztin, T., and Piribauer, P. (2022) A Bayesian approach for the estimation of weight matrices in spatial autoregressive models. *Spatial Economic Analysis*, 1-20.

Index