Package 'spflow'

October 14, 2022

Title Spatial Econometric Interaction Models

Version 0.1.0

Date 2021-09-08

Description

Efficient estimation of spatial econometric models of origin-destination flows, which may exhibit spatial autocorrelation in the dependent variable, the explanatory variables or both.
The model is the one proposed by LeSage and Pace (2008) <doi:10.1111/j.1467-9787.2008.00573.x>, who develop a matrix formulation that exploits the relational structure of flow data.
The estimation procedures follow most closely those outlined by Dargel (2021) (preprint available at <https://www.tse-fr.eu/fr/publications/revisiting-estimation-methods-spatial-econometric-interaction-models>).

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

RdMacros Rdpack

Imports coda, Matrix, methods, RSpectra, Rdpack, utils

Collate 'spflow-package.R' 'utils_classes.R'
'class_0_generics-and-maybe-classes.R'
'class_sp-network-pair.R' 'class_sp-network-nodes.R'
'class_sp-network-meta.R' 'class_sp-network-multi.R'
'class_spflow-model-meta.R' 'class_spflow-model-children.R'
'example_data.R' 'fit-spflow-mcmc.R' 'fit-spflow-mle.R'
'fit-spflow-mle_hessian.R' 'fit-spflow-ols.R'
'fit-spflow-s2sls.R' 'spflow-control.R' 'spflow-formula.R'
'spflow-likelihood.R' 'spflow-logdet.R'
'spflow-model-estimation.R' 'spflow-model-matrix_1.R'
'spflow-model-matrix_2-transformations.R'
'spflow-model-matrix_3-spatial-lags.R'
'spflow-model-moments_1.R' 'spflow-model-moments_2-blocks.R'
'spflow-simulations.R' 'spflow.R' 'utils.R' 'utils_error.R'
'utils_formula.R' 'utils_fp.R' 'utils_math.R' 'utils_matrix.R'
'utils_strings.R'

URL https://github.com/LukeCe/spflow

BugReports https://github.com/LukeCe/spflow/issues

Suggests covr, knitr, sf, sp, spdep, tinytest, rmarkdown

Language en-US

Depends R (>= 3.5.0)

VignetteBuilder knitr

ByteCompile true

NeedsCompilation no

Author Lukas Dargel [aut, cre] (<https://orcid.org/0000-0002-4839-506X>),
Thibault Laurent [aut] (<https://orcid.org/0000-0001-7487-7671>),
Christine Thomas [ths, ctb],
Paula Margaretic [ctb],
Gabriel Watkinson [ctb],
Roger Bivand [ctb]

Maintainer Lukas Dargel <lukas.dargel@mailbox.org>

Repository CRAN

Date/Publication 2021-09-09 09:40:02 UTC

R topics documented:

dat *Generic for accessing the data inside spflow network classes*

Description

For details on the methods see the documentation of the corresponding classes.

Usage

```
dat(object, ...)

dat(object) <- value
```

Arguments

object	An object belonging to the spflow network classes
...	Arguments to be passed to methods
value	A data.frame to replace existing data

Value

A the data.frame describing the set of nodes or node-pairs

See Also

spflow_network_classes()

example_data *Simulated data for stylized versions of Germany and the USA*

Description

The package uses the same stylized country examples as those presented by Yang et al. (2017). The first example is a stylized version of Germany with 16 states. The second example is a stylized version of the USA with 51 states.

We provide spatial objects that contain the geography of our example, as well as two sp_network_nodes-class() objects. Finally, there is an sp_multi_network-class() containing the two networks and four network pair objects based on the same example.

The simulation parameters are given as rho for the auto-regressive parameter, delta for the impact of the exogenous variables and sd_error for standard deviation of the simulated Gaussian noise. They are used to simulate two flow vectors y2 and y9 for each network pair inside the multi_net_usa_ge.

Usage

```
multi_net_usa_ge

germany_grid

usa_net

usa_grid

germany_net

simulation_params
```

Format

An object of class `sp_multi_network` of length 1.

An object of class `sf` (inherits from `data.frame`) with 16 rows and 3 columns.

An object of class `sp_network_nodes` of length 1.

An object of class `sf` (inherits from `data.frame`) with 51 rows and 3 columns.

An object of class `sp_network_nodes` of length 1.

An object of class `list` of length 3.

Source

Simulated data inspired by https://ialab.it.monash.edu/~dwyer/papers/maptrix.pdf

References

Yang Y, Dwyer T, Goodwin S, Marriott K (2017). "Many-to-Many Geographically-Embedded Flow Visualisation: An Evaluation." *IEEE Transactions on Visualization and Computer Graphics*, 411–420.

---

id *Generic for accessing the ids of spflow network classes*

Description

For details on the methods see the documentation of the corresponding classes.

For details see the documentation of the corresponding classes.

Usage

```
id(object, ...)

id(object, ...) <- value
```

Arguments

object	A spflow network class
...	Arguments to be passed to methods
value	A character replacing the existing id

Value

A character, corresponding to the id

See Also

spflow_network_classes()

---

mcmc_results *Access the sampling results of a* spflow_model_mcmc-class()

Description

For details see the documentation of the corresponding class.

Usage

```
mcmc_results(object)
```

Arguments

object	A spflow_model_mcmc-class()

Value

An mcmc object, containing the draws of the sampled parameters

See Also

spflow_model_mcmc-class()

neighborhood — *Generic for accessing the neighborhood matrix inside spflow network classes*

Description

For details on the methods see the documentation of the corresponding classes.

Usage

```
neighborhood(object, ...)

neighborhood(object) <- value
```

Arguments

object	One of sp_network_nodes-class(), sp_multi_network-class()
...	Arguments to be passed to methods
value	A neighborhood matrix to replace the existing one

Value

A matrix (optionally sparse), representing the neighborhood links betweem the nodes

See Also

sp_network_nodes-class() sp_multi_network-class()

nnodes — *Generic for accessing the node count of spflow network classes*

Description

For details on the methods see the documentation of the corresponding classes.

Usage

```
nnodes(object, ...)
```

Arguments

object	One of sp_network_nodes-class(), sp_network_pair-class()
...	Arguments to be passed to methods

Value

A numeric, corresponding to the number of nodes

See Also

sp_network_nodes-class() sp_network_pair-class()

---

npairs *Generic for accessing the node pairs count of a* sp_network_pair-class()

Description

For details on the method see the documentation of the corresponding class.

Usage

```
npairs(object)
```

Arguments

object	A sp_network_pair-class()

Value

A numeric, corresponding to the number of node-pairs

See Also

sp_network_pair-class()

---

pair_merge *Generic for merging information on origins and destination to node pairs inside spflow network classes*

Description

For details on the methods see the documentation of the corresponding classes.

The method merges all available information on origins and destinations to the data.frame describing the pairs.

Usage

```
pair_merge(object, ...)

## S4 method for signature 'sp_multi_network'
pair_merge(object, network_pair_id, all_pairs = FALSE)
```

Arguments

object	A sp_multi_network-class()
...	Arguments to be passed to methods
network_pair_id	A character indicating the id of a sp_network_pair-class()
all_pairs	A logical, when set to TRUE the resulting data.frame contains all possible pairs of origins and destination, even if the data in the sp_network_pair-class() does not have them.

Value

A single data.frame, combining all available information on origins, destinations and OD pairs

See Also

sp_multi_network-class()

Examples

```
# long form data for flows from Germany to Germany
pair_merge(multi_net_usa_ge,"ge_ge")

# long form data for flows from Germany to USA
pair_merge(multi_net_usa_ge,"ge_usa")
```

---

paris_data *Example data for commuting flows within Paris*

Description

The package includes an example data set that contains home-to-work commuting flows for 71 municipalities around the center of Paris. The data for the example is stored in three objects

Usage

```
paris10km_municipalities

paris10km_commuteflows

paris10km_neighborhood
```

Format

An object of class sf (inherits from data.frame) with 71 rows and 6 columns.

An object of class data.frame with 5041 rows and 4 columns.

An object of class list of length 3.

Details

1. paris10km_municipalities contains information on the municipalities. It contains some socio-economic variables, the identifier of the municipality and a geometry column. The geometry is a MULTIPOLYGON that describes the shape of the region.
2. paris10km_commuteflows contains information on pairs of municipalities. It is a data.frame with origin and destination identifiers and contains the information on the size of the commuting flows and the distance.
3. paris10km_neighborhood contains three sparse matrices that represent alternative definitions of the neighborhood of the municipalities. The first is based on contiguity, the second one on distance and the third one is based on the three nearest neighbours.

Source

The data combines different public sources (last accessed 2021-05-05).

Three data sets are provided by INSEE. More information on the rights to use and diffuse this data is provided here.

- Commuting flows (https://www.insee.fr/fr/statistiques/fichier/3566477/base-texte-flux-mobilite-domicile-lieu-travail-2015.zip)
- Number of companies (https://www.insee.fr/fr/statistiques/2021271)
- Median income (https://www.insee.fr/fr/statistiques/3560121)

The geographies, population, and area of the municipalities come from OpenDataSoft and are available here. This data set is published under an OPEN LICENCE.

---

pull_member *Generic for accessing a* sp_network_pair-class() *or a* sp_network_nodes-class() *inside a* sp_multi_network-class()

Description

For details on the method see the documentation of the corresponding class.

Usage

```
pull_member(object, ...)
```

Arguments

object	A sp_multi_network-class()
...	Arguments to be passed to methods

Value

A sp_network_pair-class() or a sp_network_nodes-class()

See Also

sp_multi_network-class()

---

results *Generic for accessing the results of a* spflow_model-class().

Description

For details on the methods see the documentation of the corresponding classes.

For details see the documentation of the corresponding class.

Usage

```
results(object)
```

Arguments

object	A spflow_model-class()

Value

A data.frame, summarizing the results of the estimation

See Also

spflow_model-class()

---

sd_error *Generic for accessing the standard deviation of the residual inside a* spflow_model-class().

Description

For details on the methods see the documentation of the corresponding classes.

For details see the documentation of the corresponding class.

Usage

```
sd_error(object)
```

Arguments

object	A spflow_model-class()

Value

A numeric, representing the estimated standard deviation of the error term

See Also

spflow_model-class()

---

spflow *Estimate spatial interaction models that incorporate spatial dependence*

Description

We implement three different estimators of spatial econometric interaction models (Dargel 2021) that allow the user to estimate origin-destination flows with spatial autocorrelation.

By default the estimation will include spatial dependence in the dependent variable and the explanatory variables which leads to the spatial Durbin model (SDM) (Anselin 1988). Moreover, the model includes an additional set of parameters for intra regional flows that start and end in the same geographic site (as proposed by LeSage and Pace (2009)). Both default options can be deactivated via the flow_control argument, which gives fine grained control over the estimation.

Usage

```
spflow(
  flow_formula,
  sp_multi_network,
  network_pair_id = id(sp_multi_network)[["network_pairs"]][[1]],
  flow_control = spflow_control()
)
```

Arguments

flow_formula	A formula specifying the spatial interaction model (for details see section Formula interface)
sp_multi_network	A sp_multi_network() object that contains information on the origins, and the destinations and their neighborhood structure
network_pair_id	A character indicating the id of a sp_network_pair() (only relevant if the sp_multi_network() contains multiple sp_network_pair-objects: defaults to the of them)
flow_control	A list generated by spflow_control() that provides fine grained control over the estimation procedure

Value

An S4 class of type spflow_model-class()

Details

Our estimation procedures makes use of the matrix formulation introduced by LeSage and Pace (2008) and further developed by Dargel (2021) to reduce the computational effort and memory requirements. The estimation procedure can be adjusted through the `estimation_method` argument in `spflow_control()`.

Maximum likelihood estimation (MLE):

Maximum likelihood estimation is the default estimation procedure. The matrix form estimation in the framework of this model was first developed by LeSage and Pace (2008) and then improved by Dargel (2021).

Spatial two-stage least squares (S2SLS):

The S2SLS estimator is an adaptation of the one proposed by Kelejian and Prucha (1998), to the case of origin-destination flows, with up to three neighborhood matrices Dargel (2021). A similar estimation is done by Tamesue and Tsutsumi (2016). The user can activate the S2SLS estimation via the `flow_control` argument using the input `spflow_control(estimation_method = "s2sls")`.

Bayesian Markov Chain Monte Carlo (MCMC):

The MCMC estimator is based on the ideas of LeSage and Pace (2009) and incorporates the improvements proposed in Dargel (2021). The estimation is based on a tuned Metropolis-Hastings sampler for the auto-regressive parameters, and for the remaining parameters it uses Gibbs sampling. The routine uses 5500 iterations of the sampling procedure and considers the first 2500 as burn-in period. The user can activate the S2SLS estimation via the `flow_control` argument using the input `spflow_control(estimation_method = "mcmc")`.

Formula interface:

The function offers a formula interface adapted to spatial interaction models, which has the following structure: `Y ~ O_(X1) + D_(X2) + I_(X3) + G_(X4)` This structure reflects the different data sources involved in such a model. On the left hand side there is the independent variable `Y` which corresponds to the vector of flows. On the right hand side we have all the explanatory variables. The functions `O_(...)` and `D_(...)` indicate which variables are used as characteristics of the origins and destinations respectively. Similarly, `I_(...)` indicates variables that should be used for the intra-regional parameters. Finally, `G_(...)` declares which variables describe origin-destination pairs, which most frequently will include a measure of distance.

All the declared variables must be available in the provided `sp_multi_network()` object, which gathers information on the origins and destinations (inside `sp_network_nodes()` objects), as well as the information on the origin-destination pairs (inside a `sp_network_pair()` object).

Using the short notation `Y ~ .` is possible and will be interpreted as usual, in the sense that we use all variables that are available for each data source. Also mixed formulas, such as `Y ~ . + G_(log(X4) + 1)`, are possible. When the dot shortcut is combined with explicit declaration, it will only be used for the non declared data sources. The following examples illustrate this behaviour.

Formula interface (examples):

Consider the case where we have the flow vector `Y` and the distance vector `DIST` available as information on origin-destination pairs. In addition we have the explanatory variables `X1, X2` and `X3` which describe the regions that are at the same time origins and destinations of the flows.

For this example the four formulas below are equivalent and make use of all explanatory variables `X1`, `X2` and `X3` for origins, destinations and intra-regional observations.

- `Y ~ .`
- `Y ~ . + G_(DIST)`
- `Y ~ X1 + X2 + X3 + G_(DIST)`
- `Y ~ D_(X1 + X2 + X3) + O_(X1 + X2 + X3) + I_(X1 + X2 + X3) + G_(DIST)`

Now if we only want to use X1 for the intra-regional model we can do the following (again all four options below are equivalent).

- `Y ~ . + I_(X1)`
- `Y ~ . + I_(X1) + G_(DIST)`
- `Y ~ X1 + X2 + X3 + I_(X1) + G_(DIST)`
- `Y ~ D_(X1 + X2 + X3) + O_(X1 + X2 + X3) + I_(X1) + G_(DIST)`

This behaviour is easily combined with transformation of variables as the two equivalent options below illustrate.

- `log(Y + 1) ~ sqrt(X1) + X2 + G_(log(DIST + 1))`

References

Anselin L (1988). *Spatial Econometrics: Methods and Models*. Springer Netherlands.

Dargel L (2021). "Revisiting Estimation Methods for Spatial Econometric Interaction Models." Toulouse School of Economics.

Kelejian HH, Prucha IR (1998). "A Generalized Spatial Two-Stage Least Squares Procedure for Estimating a Spatial Autoregressive Model with Autoregressive Disturbances." *The Journal of Real Estate Finance and Economics*, 99–121.

LeSage JP, Pace RK (2009). *Introduction to Spatial Econometrics*. CRC Press.

LeSage JP, Pace RK (2008). "Spatial Econometric Modeling of Origin-Destination Flows." *Journal of Regional Science*, 941–967.

Tamesue K, Tsutsumi M (2016). "Dealing with Intraregional Flows in Spatial Econometric Gravity Models." In Patuelli R, Arbia G (eds.), *Spatial Econometric Interaction Modelling*, chapter 6, 105–119. Springer International Publishing.

See Also

`spflow_control()` `spflow_network_classes()`

Examples

```
# Estimate flows between the states of Germany
spflow(flow_formula = y9 ~ . + G_(DISTANCE),
       sp_multi_network = multi_net_usa_ge,
       network_pair_id = "ge_ge")
```

```
# Same as above with explicit declaration of variables...
# ... X is the only variable available
# ... it is used for origins, destination and intra-state flows
spflow(flow_formula = y9 ~ X + G_(DISTANCE),
       sp_multi_network = multi_net_usa_ge,
       network_pair_id = "ge_ge")

# Same as above
spflow(flow_formula = y9 ~ O_(.) + D_(.) + I_(.) + G_(DISTANCE),
       sp_multi_network = multi_net_usa_ge,
       network_pair_id = "ge_ge")

# Same as above
spflow(flow_formula = y9 ~ O_(X) + D_(X) + I_(X) + G_(DISTANCE),
       sp_multi_network = multi_net_usa_ge,
       network_pair_id = "ge_ge")
```

---

spflow_control *Define details of the estimation procedure with the* spflow() *function.*

Description

This function creates a list to fine tune the estimation of a spatial interaction model with spflow(). The options allow to adjust the estimation method and give the user full control over the use of the explanatory variables. The user can also adjust the form of autocorrelation to be considered.

Usage

```
spflow_control(
  estimation_method = "mle",
  model = "model_9",
  use_intra = TRUE,
  sdm_variables = "same",
  weight_variable = NULL,
  parameter_space = "approx",
  loglik_det_aprox_order = 10,
  mle_hessian_method = "mixed",
  mle_optim_limit = 100,
  twosls_instrumental_variables = "same",
  twosls_decorrelate_instruments = FALSE,
  twosls_reduce_pair_instruments = TRUE,
  mcmc_iterations = 5500,
  mcmc_burn_in = 2500,
  mcmc_resampling_limit = 100
)
```

Arguments

`estimation_method`
: A character which indicates the estimation method, should be one of `c("mle","s2sls","mcmc")`

`model`
: A character indicating the model number, indicating different spatial dependence structures (see documentation for details), should be one of `paste0("model_", 1:9)`

`use_intra`
: A logical which activates the option to use a separate set of parameters for intra-regional flows (origin == destination)

`sdm_variables`
: Either a formula or a character; the formula can be used to explicitly declare the variables in SDM specification, the character should be one of `c("same", "all")` which are short cuts for using all available variables or the same as used in the main formula provided to `spflow()`

`weight_variable`
: A character indicating the name of one column in the node pair data.

`parameter_space`
: A character indicating how to define the limits of the parameter space. The only available option is `c("approx")`.

`loglik_det_aprox_order`
: A numeric indicating the order of the Taylor expansion used to approximate the value of the log-determinant term.

`mle_hessian_method`
: A character which indicates the method for Hessian calculation

`mle_optim_limit`
: A numeric indicating the number of trials given to the optimizer of the likelihood function. A trial refers to a new initiation of the optimization procedure using different (random) starting values for the parameters. If the optimizer does not converge after the indicated number of trails an error will be thrown after this limit.

`twosls_instrumental_variables`
: Either a formula or a character; the formula can be used to explicitly declare the variables that should be used as instruments during S2SLS estimation, the character should be one of `c("same", "all")` which are short cuts for using all available variables or the same as used in the main formula provided to `spflow()`

`twosls_decorrelate_instruments`
: A logical whether to perform a PCA to remove (linear) correlation from the instruments generated for the S2SLS estimator

`twosls_reduce_pair_instruments`
: A logical that indicates whether the number of instruments that are derived from pair attributes should be reduced or not. The default is `TRUE`, because constructing these instruments is often the most demanding part of the estimation (Dargel 2021).

`mcmc_iterations`
: A numeric indicating the number of iterations

`mcmc_burn_in`
: A numeric indicating the length of the burn in period

`mcmc_resampling_limit`
: A numeric indicating the maximal number of trials during rejection sampling of the autoregressive parameters

Value

A list of parameters used to control the model estimation with `spflow()`

Details

Adjusting the form of autocorrelation:

The option `model` allows to declare one of nine different forms of autocorrelation that follow the naming convention of LeSage and Pace (2008). The most general specification is "model_9", leading to the model $y = \rho_d W_d y + \rho_o W_o y + \rho_w W_w y + Z\delta + \epsilon$. All other models special cases of this one. The constraints that lead to the different sub models are summarized in this table.

Model Number	Autocorrelation Parameters	Constraints
Model 9	ρ_d, ρ_o, ρ_w	unconstrained
Model 8	ρ_d, ρ_o, ρ_w	$\rho_w = -\rho_d \rho_o$
Model 7	ρ_d, ρ_o	$\rho_w = 0$
Model 6	ρ_{dow}	$\rho_d = \rho_o = \rho_w$
Model 5	ρ_{do}	$\rho_d = \rho_o, \rho_w = 0$
Model 4	ρ_w	$\rho_d = \rho_o = 0$
Model 3	ρ_o	$\rho_d = \rho_w = 0$
Model 2	ρ_d	$\rho_o = \rho_w = 0$
Model 1	none	$\rho_d = \rho_o = \rho_w = 0$

References

Dargel L (2021). "Revisiting Estimation Methods for Spatial Econometric Interaction Models." Toulouse School of Economics.

LeSage JP, Pace RK (2008). "Spatial Econometric Modeling of Origin-Destination Flows." *Journal of Regional Science*, 941–967.

See Also

`spflow()`

Examples

```
# default is MLE estimation of the most comprehensive model
default_control <- spflow_control()

# change the estimation method
custom_control <- spflow_control(estimation_method = "mcmc")

# change the form of autocorrelation to be considered
```

```
custom_control <- spflow_control(model = "model_7")

# declare precisely which variables are to be used in the SDM form
custom_control <-
  spflow_control(sdm_variables = ~ O_(v1 + v2) + D_(v2 + v3) + I_(v1 + v4))

# deactivate the intra-regional coefficients and SDM variables
custom_control <- spflow_control(use_intra = FALSE, sdm_variables = "none")
```

---

spflow_model-class *Class spflow_model*

Description

An S4 class that contains the estimation results of spatial econometric interaction models estimated by the spflow() function.

There are four subclasses that are specific to the chosen estimation method (OLS, MLE, Bayesian MCMC or S2SLS). They contain some additional information specific to the corresponding method but most behaviours and data are identical among them.

Usage

```
## S4 method for signature 'spflow_model'
coef(object)

## S4 method for signature 'spflow_model'
fitted(object)

## S4 method for signature 'spflow_model'
nobs(object)

## S4 method for signature 'spflow_model'
predict(object, ..., type = "BP")

## S4 method for signature 'spflow_model'
resid(object)

## S4 method for signature 'spflow_model'
results(object)

## S4 method for signature 'spflow_model'
sd_error(object)

## S4 method for signature 'spflow_model_mle'
logLik(object)

## S4 method for signature 'spflow_model_mle_s2sls_ols'
```

```
varcov(object)

## S4 method for signature 'spflow_model_mcmc'
mcmc_results(object)
```

Arguments

object	A spflow_model-class()
...	Further arguments passed to the prediction function
type	A character declaring the type of prediction (for now only "BP")

Slots

estimation_results A data.frame that contains the main results() of the estimation

estimation_control A list that contains all control parameters of the estimation (see spflow_control())

N A numeric that corresponds to the number of origin-destination pairs (the sample size in this model)

sd_error A numeric representing the standard deviation of the residual

R2_corr A numeric that serves as a goodness of fit criterion. The R2_corr is computed as squared correlation between the fitted values and the observed values of the dependent variable.

resid A numeric vector of regression residuals

fitted A numeric vector of fitted values computed as the in sample prediction trend signal (TS) prediction described by @Goulard2017

spatial_filter_matrix A matrix (can be sparse) or NULL

design_matrix A matrix (can be sparse) or NULL

varcov A matrix or NULL

ll A numeric or NULL

AIC A numeric or NULL

BIC A numeric or NULL

mcmc_results A data.frame containing the estimated parameters for each iteration of the MCMC sampling procedure

Main results

The main results are accessed with the results() method. They are given in the form of a data frame with the following columns;

- est: value of the estimated parameter
- sd: value of the standard deviation of the parameter
- t.test: value of the t-statistic under the two-sided hypothesis that the parameter value is 0.
- p.val: the p-value associated to the t-test
- quant_025: for Bayesian estimation the lower bound of 95% interval
- quant_975: for Bayesian estimation the upper bound of 95% interval

See Also

spflow(), spflow_network_classes()

Examples

```
spflow_results <- spflow(y9 ~ . + G_(DISTANCE),multi_net_usa_ge)

# General methods
results(spflow_results) # data.frame of main results
coef(spflow_results) # vector of estimated coefficients
fitted(spflow_results) # vector of fitted values
resid(spflow_results) # vector of residuals
nobs(spflow_results) # number of observations
sd_error(spflow_results) # standard deviation of the error term
predict(spflow_results) # computation of the in sample predictor

# MLE methods
logLik(spflow_results) # value of the likelihood function

# MLE, OLS and S2SLS methods
varcov(spflow_results) # variance covariance matrix of the estimators

# MCMC methods
spflow_results_mcmc <- spflow(
  y2 ~ . + G_(DISTANCE),
  multi_net_usa_ge,
  flow_control = spflow_control(estimation_method = "mcmc",
                                model = "model_2"))
results(spflow_results)
mcmc_results(spflow_results_mcmc) # parameter values during the mcmc sampling
```

---

spflow_network_classes

spflow network classes

Description

The spflow package provides three additional classes to the R environment that allow to handle origin-destination flow data efficiently.

The main idea is to exploit the relational structure of origin-destination data to reduce the memory requirements. Data on origins and destinations are stored in the sp_network_nodes-class() and data on the origin-destination pairs are stored in an sp_network_pair-class().

A third object of type sp_multi_network-class() is then used to store information on the nodes and pairs in an efficient relational storage. It makes sure that all origin-destination pairs can be identified with the nodes at the origin and destination.

See Also

Other spflow network classes: sp_multi_network-class, sp_network_nodes-class, sp_network_pair-class

Other Constructors for spflow network classes: sp_multi_network, sp_network_nodes, sp_network_pair

Examples

```
### An example use case for the spflow network classes and model estimation
# load example data
data("paris10km_municipalities")
data("paris10km_neighborhood")
data("paris10km_commuteflows")

# define the sp_network_nodes...
# ... they are used as origins and destinations
# ... their neighborhood is based on contiguity
paris10km_net <- sp_network_nodes(
  network_id = "paris10km",
  node_neighborhood = paris10km_neighborhood$by_contiguity,
  node_data = sf::st_drop_geometry(paris10km_municipalities),
  node_key_column = "ID_MUN")

# define the sp_network_pair...
# ... contains pairwise data (flows and distances)
# ... must be linked to an origin and a destination network
paris10km_net_pairs <- sp_network_pair(
  orig_net_id = "paris10km",
  dest_net_id = "paris10km",
  pair_data = paris10km_commuteflows,
  orig_key_column = "ID_ORIG",
  dest_key_column = "ID_DEST")


# define the sp_network_pair...
# ... combines information on nodes and pairs
paris10km_multi_net <- sp_multi_network(paris10km_net,paris10km_net_pairs)

clog <- function(x) {
  y <- log(x)
  y - mean(y)
}

# define the model that we use to explain the flows...
# ... D_() contains destination variables
# ... O_() contains origin variables
# ... D_() contains intra-regional variables (when origin == destination)
# ... G_() contains pair variables (distances)
flow_formula <-
  log(COMMUTE_FLOW + 1) ~
  D_(log(NB_COMPANY) + clog(MED_INCOME)) +
  O_(log(POPULATION) + log(NB_COMPANY) + clog(MED_INCOME)) +
  I_(log(NB_COMPANY) + log(POPULATION)) +
```

```
  G_(log(DISTANCE + 1))

# define what variables to use in an SDM specification
# ... if not given all will be used
sdm_formula <-
 ~ D_(log(NB_COMPANY) + clog(MED_INCOME))

# define the list of control parameters
flow_control <- spflow_control(sdm_variables = sdm_formula)


# Estimate the model
spflow(flow_formula, paris10km_multi_net, flow_control = flow_control)
```

`sp_multi_network` *Create an S4 class that contains* `sp_network_nodes()` *and* `sp_network_pair()` *for one or multiple networks*

Description

Create an S4 class that contains `sp_network_nodes()` and `sp_network_pair()` for one or multiple networks

Usage

```
sp_multi_network(...)
```

Arguments

`...`	objects of type `sp_network_nodes()` and `sp_network_pair()`

Value

An S4 class of type `sp_multi_network-class()`

See Also

Other Constructors for spflow network classes: `sp_network_nodes`, `sp_network_pair`, `spflow_network_classes`

Examples

```
sp_multi_network() # empty
sp_multi_network(germany_net,usa_net) # two networks, no pairs
```

sp_multi_network-class *Class sp_multi_network*

Description

A S4 class that gathers information on one or multiple networks sp_network_nodes() and origin-destination pairs sp_network_pair(). The class is constructed with the sp_multi_network() function.

Usage

```
## S4 method for signature 'sp_multi_network'
id(object)

## S4 method for signature 'sp_multi_network'
dat(object, .id)

## S4 method for signature 'sp_multi_network'
neighborhood(object, .id = 1)

## S4 method for signature 'sp_multi_network'
pull_member(object, .id)
```

Arguments

object	sp_multi_network-class
.id	A character indicating the id of a sp_network_nodes-class() or a sp_network_pair-class() inside the sp_multi_network-class().

Slots

networks A list of sp_network_nodes-class() objects

network_pairs A list of sp_network_pair-class() objects

See Also

Other spflow network classes: sp_network_nodes-class, sp_network_pair-class, spflow_network_classes

Examples

```
## access the id of a networks or network_pairs inside a multi network

id(multi_net_usa_ge)$networks
id(multi_net_usa_ge)$network_pairs

## access the data of a network or a network_pair inside a multi_network
```

```
dat(multi_net_usa_ge, "ge")    # extract data of nodes
dat(multi_net_usa_ge, "ge_ge") # extract data of pairs

## access sp_network_nodes or sp_network_pair inside a sp_multi_network

pull_member(multi_net_usa_ge,"ge")
pull_member(multi_net_usa_ge,"usa")
pull_member(multi_net_usa_ge,"ge_ge")
```

---

sp_network_nodes *Create a* sp_network_nodes-class()

Description

Create a sp_network_nodes-class()

Usage

```
sp_network_nodes(
  network_id,
  node_neighborhood = NULL,
  node_data = NULL,
  node_key_column
)
```

Arguments

network_id	A character that serves as an identifier for the network
node_neighborhood	A matrix that describes the neighborhood of the nodes
node_data	A data.frame that contains all information describing the nodes
node_key_column	A character indicating the column containing the identifiers for the nodes

Value

An S4 class of type sp_network_nodes-class()

See Also

Other Constructors for spflow network classes: sp_multi_network, sp_network_pair, spflow_network_classes

Examples

```
sp_network_nodes("germany",
                 spdep::nb2mat(spdep::poly2nb(germany_grid)),
                 as.data.frame(germany_grid),
                 "ID_STATE")
```

sp_network_nodes-class

sp_network_nodes Class

Description

An S4 class that contains all information on a single network. In this representation a network is composed of nodes which must be identified uniquely by and ID. Each node is described by variables stored in a data.frame. The node neighborhood matrix describes strength of links between the nodes of the network. The class is constructed by the sp_network_nodes() function.

Usage

```
## S4 method for signature 'sp_network_nodes'
dat(object)

## S4 replacement method for signature 'sp_network_nodes'
dat(object) <- value

## S4 method for signature 'sp_network_nodes'
id(object)

## S4 replacement method for signature 'sp_network_nodes'
id(object) <- value

## S4 method for signature 'sp_network_nodes'
neighborhood(object)

## S4 replacement method for signature 'sp_network_nodes'
neighborhood(object) <- value

## S4 method for signature 'sp_network_nodes'
nnodes(object)
```

Arguments

object	A sp_network_nodes-class
value	An object to replace the existing id/data/neighborhood

Slots

network_id A character that serves as an identifier for the network

nnodes A numeric that indicates the number of nodes in the network

node_data A data.frame that contains all information describing the nodes

node_neighborhood A matrix that describes the neighborhood relations of the nodes

See Also

Other spflow network classes: sp_multi_network-class, sp_network_pair-class, spflow_network_classes

Examples

```
## access the data describing the nodes
new_dat <- dat(germany_net)

# access the id of the network
germany_net2 <- germany_net
id(germany_net2)
id(germany_net2) <- "Germany"

# access the neighborhood matrix of the nodes
neighborhood(germany_net)

# access the number of nodes inside the network
nnodes(germany_net)
```

---

sp_network_pair *Create an S4 object that contains information on origin-destination pairs*

Description

Create an S4 object that contains information on origin-destination pairs

Usage

```
sp_network_pair(
  orig_net_id,
  dest_net_id,
  pair_data = NULL,
  orig_key_column,
  dest_key_column
)
```

Arguments

orig_net_id	A character that serves as identifier for the origin network
dest_net_id	A character that serves as identifier for the destination network
pair_data	A data.frame containing information on the origin-destination pairs
orig_key_column	A character indicating the name of the column containing the identifiers of the origins
dest_key_column	A character indicating the name of the column containing the identifiers of the destinations

Value

An S4 class of type sp_network_pair-class()

See Also

Other Constructors for spflow network classes: sp_multi_network, sp_network_nodes, spflow_network_classes

Examples

```
pair_frame <- data.frame(
  ORIG_ID_STATE = rep(germany_grid$ID_STATE, times = 16),
  DEST_ID_STATE = rep(germany_grid$ID_STATE, each = 16))
sp_network_pair("ge","ge",pair_frame,"ORIG_ID_STATE","DEST_ID_STATE")
```

---

sp_network_pair-class *Class sp_network_pair*

Description

An S4 class which holds information on origin-destination (OD) pairs. Each OD pair is composed of two nodes (see sp_network_nodes-class()). All origins belong to the same (origin-) network and all destination belong to the same (destination-) network. It is possible to choose the same network for origins and destinations, which enables to represent OD pairs within the same network.

Usage

```
## S4 method for signature 'sp_network_pair'
dat(object)

## S4 replacement method for signature 'sp_network_pair'
dat(object) <- value

## S4 method for signature 'sp_network_pair'
id(object)

## S4 replacement method for signature 'sp_network_pair'
id(object) <- value

## S4 method for signature 'sp_network_pair'
npairs(object)

## S4 method for signature 'sp_network_pair'
nnodes(object)
```

Arguments

object	A sp_network_pair-class
value	An object to replace the existing id/data

Slots

`orig_net_id` A character that serves as identifier for the origin network

`orig_nnodes` A numeric that represents the number of nodes in the origin network

`dest_net_id` A character that serves as identifier for the destination network

`dest_nnodes` A numeric that represents the number of nodes in the destination network

`network_pair_id` A character identifying the pair of networks

`pair_data` A data.frame containing information on origin-destination pairs

`npairs` A numeric indicating the number of origin-destination pairs

See Also

Other spflow network classes: `sp_multi_network-class`, `sp_network_nodes-class`, `spflow_network_classes`

Examples

```
## access the data describing the node pairs

net_pair_ge_ge <- pull_member(multi_net_usa_ge,"ge_ge")
dat(net_pair_ge_ge)

## access the id of a network pair

net_pair_ge_ge <- pull_member(multi_net_usa_ge,"ge_ge")
id(net_pair_ge_ge)
id(net_pair_ge_ge) <- "Germany_Germany"

## access the number of node pairs in a network pair

net_pair_ge_ge <- pull_member(multi_net_usa_ge,"ge_ge")
npairs(net_pair_ge_ge)

## access the number of origin and destination nodes in a network pair
net_pair_ge_ge <- pull_member(multi_net_usa_ge,"ge_ge")
nnodes(net_pair_ge_ge)
nnodes(net_pair_ge_ge)["orig"]
nnodes(net_pair_ge_ge)["dest"]
prod(nnodes(net_pair_ge_ge) == npairs(net_pair_ge_ge))
```

---

varcov *Generic for accessing the variance-covariance matrix of the parameters inside a* `spflow_model-class()`.

Description

For details on the methods see the documentation of the corresponding classes.

Usage

```
varcov(object)
```

Arguments

object	A spflow_model-class()

Value

The variance-covariance matrix of the fitted model parameters

See Also

spflow_model-class()

Index