Package ‘spfilteR’

May 13, 2021

Type Package

Title Semiparametric Spatial Filtering with Eigenvectors in (Generalized) Linear Models

Version 1.1.0

Description Tools to decompose (transformed) spatial connectivity matrices and perform supervised or unsupervised semiparametric spatial filtering in a regression framework. The package supports unsupervised spatial filtering in standard linear as well as some generalized linear regression models.

License GPL-3

URL https://github.com/sjuhl/spfilteR

BugReports https://github.com/sjuhl/spfilteR/issues

Depends R (>= 3.5.0)

Imports stats

Suggests testthat, knitr, rmarkdown

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

VignetteBuilder knitr

NeedsCompilation no

Author Sebastian Juhl [aut, cre] (<https://orcid.org/0000-0002-7123-5398>)

Maintainer Sebastian Juhl <sebastian.juhl@gess.uni-mannheim.de>

Repository CRAN

Date/Publication 2021-05-12 23:00:08 UTC

R topics documented:

  fakedata ................................................................. 2
  getEVs ................................................................. 3
  glmFilter ............................................................. 4
**Description**

An artificially generated cross-sectional dataset together with an accompanying binary connectivity matrix $W$. The $n = 100$ units are located on a regular grid and $W$ is defined according to rook’s adjacency definition of contiguity. The synthetic data can be used to illustrate the functionality of this package.

**Usage**

```r
data(fakedata)

W
```

**Format**

An object of class `data.frame` with 100 rows and 8 columns.

An object of class `matrix` (inherits from `array`) with 100 rows and 100 columns.

**Value**

The file contains two objects:

- `fakedataset` a synthetic dataset
- `W` an artificial spatial connectivity matrix

**Examples**

```r
data(fakedata)
head(fakedataset)
dim(W)
```
Description

Extract eigenvectors and corresponding eigenvalues from the matrix $MWM$, where $M$ denotes a symmetric and idempotent projection matrix and $W$ is the spatial connectivity matrix. This function also reports the Moran coefficient associated with each of the eigenvectors.

Usage

```r
getEVs(W, covars = NULL)
```

Arguments

- `W`: spatial connectivity matrix
- `covars`: vector/ matrix of regressors included in the construction of the projection matrix $M$ - see Details

Details

The eigenfunctions obtained by `getEVs` can be used to perform supervised eigenvector selection and to manually create a spatial filter. To this end, a candidate set may be determined by 1) the sign of the spatial autocorrelation in model residuals and 2) the strength of spatial association found in each eigenvector as indicated by `moran`.

Prior to the spectral decomposition, `getEVs` symmetrizes the spatial connectivity matrix by: $\frac{1}{2} * (W + W')$.

If `covars` are supplied, the function uses the covariates to construct projection matrix: $M = I - X (X'X)^{-1} X'$. Using this matrix results in a set of eigenvectors that are uncorrelated to each other as well as to the covariates. If `covars=NULL`, only the intercept term is used to construct $M$. See e.g., Griffith and Tiefelsdorf (2007) for more details on the appropriate choice of $M$.

Value

A list containing the following objects:

- `vectors`: matrix of all eigenvectors
- `values`: vector of the corresponding eigenvalues
- `moran`: vector of the Moran coefficients associated with the eigenvectors

Author(s)

Sebastian Juhl
References

See Also
lmFilter, glmFilter, MI,ev, MI.sf, vif.ev, partialR2

Examples
```r
data(fakedata)
E <- getEVs(W=W, covars=NULL)
```

glmFilter

Unsupervised Spatial Filtering with Eigenvectors in Generalized Linear Regression Models

Description
This function implements the eigenvector-based semiparametric spatial filtering approach in a generalized linear regression framework using maximum likelihood estimation (MLE). Eigenvectors are selected by an unsupervised stepwise regression technique. Supported selection criteria are the minimization of residual autocorrelation, maximization of model fit, significance of residual autocorrelation, and the statistical significance of eigenvectors. Alternatively, all eigenvectors in the candidate set can be included as well.

Usage
```r
glmFilter(
y, x = NULL, W, objfn = "AIC", MX = NULL, model, optim.method = "BFGS", sig = 0.05, bonferroni = TRUE, positive = TRUE, ideal.setsize = FALSE, min.reduction = 0.05, boot.MI = 100, resid.type = "pearson", alpha = 0.25,
```
glmFilter

tol = 0.1,
na.rm = TRUE

Arguments

Arguments are:

- **y**: response variable
- **x**: vector/matrix of regressors (default=NULL)
- **W**: spatial connectivity matrix
- **objfn**: the objective function to be used for eigenvector selection. Possible criteria are: the maximization of model fit ('AIC' or 'BIC'), minimization of residual autocorrelation ('MI'), significance level of candidate eigenvectors ('p'), significance of residual spatial autocorrelation ('pMI'), or all eigenvectors in the candidate set ('all')
- **MX**: covariates used to construct the projection matrix (default=NULL) - see Details
- **model**: a character string indicating the type of model to be estimated. Currently, 'probit', 'logit', and 'poisson' are valid inputs
- **optim.method**: a character specifying the optimization method used by the optim function
- **sig**: significance level to be used for eigenvector selection if objfn='p' or objfn='pMI'
- **bonferroni**: Bonferroni adjustment for the significance level (TRUE/FALSE) if objfn='p'. Set to FALSE if objfn='pMI' - see Details
- **positive**: restrict search to eigenvectors associated with positive levels of spatial autocorrelation (TRUE/FALSE)
- **ideal.setsize**: if positive=TRUE, uses the formula proposed by Chun et al. (2016) to determine the ideal size of the candidate set (TRUE/FALSE)
- **min.reduction**: if objfn is either 'AIC' or 'BIC'. A value in the interval [0,1) that determines the minimum reduction in AIC/BIC (relative to the current AIC/BIC) a candidate eigenvector need to achieve in order to be selected
- **boot.MI**: number of iterations used to estimate the variance of Moran’s I (default=100). Alternatively, if boot.MI=NULL, analytical results will be used
- **resid.type**: character string specifying the residual type to be used. Options are 'raw', 'deviance', and 'pearson' (default)
- **alpha**: a value in (0,1] indicating the range of candidate eigenvectors according to their associated level of spatial autocorrelation, see e.g., Griffith (2003)
- **tol**: if objfn='MI', determines the amount of remaining residual autocorrelation at which the eigenvector selection terminates
- **na.rm**: remove observations with missing values (TRUE/FALSE)

Details

If **W** is not symmetric, it gets symmetrized by 1/2 * (W + W') before the decomposition.

If covariates are supplied to **MX**, the function uses these regressors to construct the following projection matrix:
\[ M = I - X (X'X)^{-1}X' \]

Eigenvalues from \( MWM \) using this specification of \( M \) are not only mutually uncorrelated but also orthogonal to the regressors specified in \( MX \). Alternatively, if \( MX=NULL \), the projection matrix becomes \( M = I - \frac{1}{n}11' \), where \( I \) is a vector of ones and \( n \) represents the number of observations. Griffith and Tiefelsdorf (2007) show how the choice of the appropriate \( M \) depends on the underlying process that generates the spatial dependence.

The Bonferroni correction is only possible if eigenvector selection is based on the significance level of the eigenvectors (\( \text{objfn} = 'p' \)). It is set to FALSE if eigenvectors are added to the model until the residuals exhibit no significant level of spatial autocorrelation (\( \text{objfn} = 'pMI' \)).

**Value**

An object of class `spfilter` containing the following information:

- `estimates` summary statistics of the parameter estimates
- `varcovar` estimated variance-covariance matrix
- `EV` a matrix containing the summary statistics of selected eigenvectors
- `selvecs` vector/matrix of selected eigenvectors
- `evMI` Moran coefficient of all eigenvectors
- `moran` residual autocorrelation in the initial and the filtered model
- `fit` adjusted R-squared of the initial and the filtered model
- `residuals` initial and filtered model residuals
- `other` a list providing supplementary information:
  - `ncandidates` number of candidate eigenvectors considered
  - `nev` number of selected eigenvectors
  - `condnum` condition number to assess the degree of multicollinearity among the eigenvectors induced by the link function, see e.g., Griffith/ Amrhein (1997)
  - `sel_id` ID of selected eigenvectors
  - `sf` vector representing the spatial filter
  - `sfMI` Moran coefficient of the spatial filter
  - `model` type of the regression model
  - `dependence` filtered for positive or negative spatial dependence
  - `objfn` selection criterion specified in the objective function of the stepwise regression procedure
  - `bonferroni` TRUE/ FALSE: Bonferroni-adjusted significance level (if \( \text{objfn} = 'p' \))
  - `siglevel` if \( \text{objfn} = 'p' \) or \( \text{objfn} = 'pMI' \): actual (unadjusted/ adjusted) significance level
  - `resid.type` residual type (‘raw’, ‘deviance’, or ‘pearson’)
  - `pseudoR2` McFadden’s pseudo R-squared (filtered vs. unfiltered model)

**Note**

If the condition number (\( \text{condnum} \)) suggests high levels of multicollinearity, eigenvectors can be sequentially removed from `selvecs` and the model can be re-estimated using the `glm` function in order to identify and manually remove the problematic eigenvectors. Moreover, if other models that
are currently not implemented here need to be estimated (e.g., quasi-binomial models), users can
extract eigenvectors using the function `getEVs` and perform a supervised eigenvector search using
the `glm` function.

In contrast to eigenvector-based spatial filtering in linear regression models, Chun (2014) notes
that only a limited number of studies address the problem of measuring spatial autocorrelation
in generalized linear model residuals. Consequently, eigenvector selection may be based on an
objective function that maximizes model fit rather than minimizes residual spatial autocorrelation.

References


See Also

`lmFilter`, `getEVs`, `MI.resid`, `optim`

Examples

data(fakedata)

# poisson model
y_pois <- fakedataset$count
poisson <- glmFilter(y=y_pois,x=NULL,W=W,objfn="MI",positive=FALSE,
                    model="poisson",boot.MI=100)
print(poisson)
summary(poisson,EV=FALSE)

# probit model - summarize EVs
y_prob <- fakedataset$indicator
probit <- glmFilter(y=y_prob,x=NULL,W=W,objfn="p",positive=FALSE,
                    model="probit",boot.MI=100)
print(probit)
summary(probit,EV=TRUE)

# logit model - AIC objective function
y_logit <- fakedataset$indicator
logit <- glmFilter(y=y_logit,x=NULL,W=W,objfn="AIC",positive=FALSE,
                   model="logit",min.reduction=.05)
print(logit)
summary(logit,EV=FALSE)
Description

This function implements the eigenvector-based semiparametric spatial filtering approach in a linear regression framework using ordinary least squares (OLS). Eigenvectors are selected by an unsupervised stepwise regression technique. Supported selection criteria are the minimization of residual autocorrelation, maximization of model fit, significance of residual autocorrelation, and the statistical significance of eigenvectors. Alternatively, all eigenvectors in the candidate set can be included as well.

Usage

```r
lmFilter(
  y,
  x = NULL,
  W,
  objfn = "MI",
  MX = NULL,
  sig = 0.05,
  bonferroni = TRUE,
  positive = TRUE,
  ideal.setsize = FALSE,
  alpha = 0.25,
  tol = 0.1,
  boot.MI = NULL,
  na.rm = TRUE
)
```

S3 method for class 'spfilter'
```
summary(object, EV = FALSE, ...)
```

Arguments

- `y` response variable
- `x` vector/ matrix of regressors (default=NULL)
- `W` spatial connectivity matrix
- `objfn` the objective function to be used for eigenvector selection. Possible criteria are: the maximization of the adjusted R-squared ('R2'), minimization of residual autocorrelation ('MI'), significance level of candidate eigenvectors ('p'), significance of residual spatial autocorrelation ('pMI') or all eigenvectors in the candidate set ('all')
- `MX` covariates used to construct the projection matrix (default=NULL) - see Details
- `sig` significance level to be used for eigenvector selection if `objfn='p'` or `objfn='pMI'`
**bonferroni**  
Bonferroni adjustment for the significance level (TRUE/FALSE) if `objfn='p'`. Set to FALSE if `objfn='pMI'` - see Details

**positive**  
restrict search to eigenvectors associated with positive levels of spatial autocorrelation (TRUE/FALSE)

**ideal.setsize**  
if `positive=TRUE`, uses the formula proposed by Chun et al. (2016) to determine the ideal size of the candidate set (TRUE/FALSE)

**alpha**  
a value in (0,1] indicating the range of candidate eigenvectors according to their associated level of spatial autocorrelation, see e.g., Griffith (2003)

**tol**  
if `objfn='MI'`, determines the amount of remaining residual autocorrelation at which the eigenvector selection terminates

**boot.MI**  
number of iterations used to estimate the variance of Moran’s I. If `boot.MI=NULL` (default), analytical results will be used

**na.rm**  
remove observations with missing values (TRUE/FALSE)

**object**  
an object of class `spfilter`

**EV**  
display summary statistics for selected eigenvectors (TRUE/FALSE)

...  
additional arguments

**Details**

If `W` is not symmetric, it gets symmetrized by $1/2 \times (W + W')$ before the decomposition.

If covariates are supplied to `MX`, the function uses these regressors to construct the following projection matrix:

$$\mathbf{M} = \mathbf{I} - \mathbf{X} (\mathbf{X}' \mathbf{X})^{-1} \mathbf{X}'$$

Eigenvectors from `MWM` using this specification of `M` are not only mutually uncorrelated but also orthogonal to the regressors specified in `MX`. Alternatively, if `MX=NULL`, the projection matrix becomes $\mathbf{M} = \mathbf{I} - \mathbf{1}\mathbf{1}'/n$, where $\mathbf{1}$ is a vector of ones and $n$ represents the number of observations. Griffith and Tiefelsdorf (2007) show how the choice of the appropriate $\mathbf{M}$ depends on the underlying process that generates the spatial dependence.

The Bonferroni correction is only possible if eigenvector selection is based on the significance level of the eigenvectors (`objfn='p'`). It is set to FALSE if eigenvectors are added to the model until the residuals exhibit no significant level of spatial autocorrelation (`objfn='pMI'`).

**Value**

An object of class `spfilter` containing the following information:

- **estimates** summary statistics of the parameter estimates
- **varcovar** estimated variance-covariance matrix
- **EV** a matrix containing the summary statistics of selected eigenvectors
- **selvecs** vector/ matrix of selected eigenvectors
- **evMI** Moran coefficient of all eigenvectors
- **moran** residual autocorrelation in the initial and the filtered model
- **fit** adjusted R-squared of the initial and the filtered model
residuals initial and filtered model residuals
other a list providing supplementary information:
   nCandidates number of candidate eigenvectors considered
   nev number of selected eigenvectors
   sel_id ID of selected eigenvectors
   sf vector representing the spatial filter
   sfMI Moran coefficient of the spatial filter
   model type of the fitted regression model
   dependence filtered for positive or negative spatial dependence
   objfn selection criterion specified in the objective function of the stepwise regression procedure
   bonferroni TRUE/ FALSE: Bonferroni-adjusted significance level (if objfn='p')
   siglevel if objfn='p' or objfn='pMI': actual (unadjusted/ adjusted) significance level

References


See Also
glmFilter, getEVs, MI.resid

Examples

data(fakedata)
y <- fakedataset$x1
X <- cbind(fakedataset$x2,fakedataset$x3,fakedataset$x4)

res <- lmFilter(y=y,x=X,W=W,objfn='MI',positive=FALSE)
print(res)
summary(res,EV=TRUE)

E <- res$selvecs
(ols <- coef(lm(y~X+E)))
coef(res)
MI.decomp  Decomposition of the Moran Coefficient

Description
A decomposition of the Moran coefficient in order to separately test for the simultaneous presence of positive and negative autocorrelation in a variable.

Usage
MI.decomp(x, W, nsim = 100)

Arguments
- x: a vector or matrix
- W: spatial connectivity matrix
- nsim: number of iterations to simulate the null distribution

Details
If x is a matrix, this function computes the Moran test for spatial autocorrelation for each column. The $p$-values calculated for $I^+$ and $I^-$ assume a directed alternative hypothesis. Statistical significance is assessed using a permutation procedure to generate a simulated null distribution.

Value
Returns a data.frame that contains the following information for each variable:
- $I^+$: observed value of Moran’s I (positive part)
- $\text{Var} I^+$: variance of Moran’s I (positive part)
- $p I^+$: simulated $p$-value of Moran’s I (positive part)
- $I^-$: observed value of Moran’s I (negative part)
- $\text{Var} I^-$: variance of Moran’s I (negative part)
- $p I^-$: simulated $p$-value of Moran’s I (negative part)
- $p I$\_two.sided: simulated $p$-value of the two-sided test

Author(s)
Sebastian Juhl

References
See Also

MI.vec, MI.ev, MI.sf, MI.resid, MI.local, getEVs

Examples

data(fakedata)
X <- cbind(fakedataset$x1, fakedataset$x2,
fakedataset$x3, fakedataset$negative)

(MI.dec <- MI.decomp(x=X, W=W, nsim=100))

# the sum of I+ and I- equals the observed Moran coefficient:
I <- MI.vec(x=X, W=W)[, "I"]
cbind(MI.dec[, "I+"] + MI.dec[, "I-"], I)

---

**MI.ev**

*Moran Coefficients of Eigenvectors*

**Description**

Calculates the Moran coefficient for each eigenvector.

**Usage**

`MI.ev(W, evals)`

**Arguments**

- `W` : spatial connectivity matrix
- `evals` : vector of eigenvalues

**Value**

Returns a vector containing the Moran coefficients of the eigenvectors associated with the supplied eigenvalues.

**Author(s)**

Sebastian Juhl

**References**


MI.resid

Moran Test for Residual Spatial Autocorrelation

Description
This function assesses the degree of spatial autocorrelation present in regression residuals by means of the Moran coefficient.

Usage
MI.resid(resid, x = NULL, W, alternative = "greater", boot = NULL)

Arguments
- resid: residual vector
- x: vector/ matrix of regressors (default=NULL)
- W: spatial connectivity matrix
- alternative: specification of alternative hypothesis as 'greater' (default), 'lower', or 'two.sided'
- boot: optional integer specifying the number of simulation iterations to compute the variance. If NULL (default), variance calculated under assumed normality

Details
The function assumes an intercept-only model if x=NULL. Furthermore, MI.resid automatically symmetrizes the matrix W by: 1/2 * (W + W').

Value
A data.frame object with the following elements:
- I: observed value of the Moran coefficient
- EI: expected value of Moran's I
- VarI: variance of Moran's I
- zI: standardized Moran coefficient
- pI: p-value of the test statistic

Note
Calculations are based on the procedure proposed by Cliff and Ord (1981). See also Cliff and Ord (1972).
MI.sf

Moran Coefficient of the Spatial Filter

Description

Computes the Moran coefficient of the spatial filter.

Usage

MI.sf(gamma, evMI)

Arguments

gamma vector of regression coefficients associated with the eigenvectors

evMI Moran coefficient of eigenvectors
**MI.vec**

**Value**

Moran coefficient of the spatial filter.

**Author(s)**

Sebastian Juhl

**References**


**See Also**

`lmFilter`, `glmFilter`, `getEVs`, `MI.ev`

---

**Description**

Reports the local Moran Coefficient for each unit.

Tests for the presence of spatial autocorrelation in variables as indicated by the Moran coefficient. The variance is calculated under the normality assumption.

**Usage**

```r
MI.local(x, W, alternative = "greater")

MI.vec(x, W, alternative = "greater", symmetrize = TRUE)
```

**Arguments**

- `x`: a vector or matrix
- `W`: spatial connectivity matrix
- `alternative`: specification of alternative hypothesis as 'greater' (default), 'lower', or 'two.sided'
- `symmetrize`: symmetrizes the connectivity matrix `W` by: 1/2 * (W + W') (TRUE/ FALSE).

**Details**

If `x` is a matrix, this function computes the Moran test for spatial autocorrelation for each column.
Value

Returns an object of class `data.frame` that contains the following information for each variable:

- `Ii` observed value of local Moran's I
- `EIi` expected value of local Moran coefficients
- `VarIi` variance of local Moran's I
- `zIi` standardized local Moran coefficient
- `pIi` \( p \)-value of the test statistic

Returns an object of class `data.frame` that contains the following information for each variable:

- `I` observed value of the Moran coefficient
- `EI` expected value of Moran’s I
- `VarI` variance of Moran’s I (under normality)
- `zI` standardized Moran coefficient
- `pI` \( p \)-value of the test statistic

Note

The calculation of the statistic and its moments follows Anselin (1995) and Sokal et al. (1998). Estimation of the variance (under the normality assumption) follows Cliff and Ord (1981), see also Upton and Fingleton (1985). It assumes the connectivity matrix \( W \) to be symmetric. For inherently non-symmetric matrices, it is recommended to specify `symmetrize=TRUE`.

Author(s)

Sebastian Juhl
Sebastian Juhl

References


partialR2

See Also
    MI.vec, MI.ev, MI.sf, MI.resid, MI.decomp
    MI.resid, MI.local

Examples
    data(fakedata)
    x <- fakedataset$x2
    (MIi <- MI.local(x=x, W=W, alternative="greater"))
    data(fakedata)
    X <- cbind(fakedataset$x1, fakedataset$x2, fakedataset$x3)
    (MI <- MI.vec(x=X, W=W, alternative="greater", symmetrize=TRUE))

---

partialR2  

Coefficient of Partial Determination

Description
    This function computes the partial R-squared of all selected eigenvectors in a spatially filtered linear
    regression model.

Usage
    partialR2(y, x = NULL, evecs)

Arguments
    y  response variable
    x  vector/ matrix of regressors
    evecs (selected) eigenvectors

Value
    Vector of partial R-squared values of the eigenvectors.

Note
    The function assumes a linear regression model. Since the eigenvectors are mutually uncorrelated, partialR2
    evaluates them sequentially. In generalized linear models, the presence of a link function can corrupt the
    uncorrelatedness of the eigenvectors.

Author(s)
    Sebastian Juhl
See Also

`lmFilter`, `getEVs`

Examples

```r
data(fakedata)
y <- fakedataset$x1
x <- fakedataset$x2

# get eigenvectors
E <- getEVs(W=W, covars=NULL)$vectors

(out <- partialR2(y=y, x=x, evecs=E[, 1:5]))
```

---

**vif.ev**

*Variance Inflation Factor of Eigenvectors*

**Description**

Calculate the variance inflation factor (VIF) of the eigenvectors in the spatial filter.

**Usage**

```r
vif.ev(x = NULL, evecs, na.rm = TRUE)
```

**Arguments**

- `x` vector/matrix of regressors (default=NULL)
- `evecs` (selected) eigenvectors
- `na.rm` remove missing values in covariates (TRUE/ FALSE)

**Value**

Returns a vector containing the VIF for each eigenvector.

**Note**

This function assumes a linear model which ensures the uncorrelatedness of the eigenvectors. Note that regression weights or the link function used in generalized linear models can corrupt this property.

**Author(s)**

Sebastian Juhl
Variance Partitioning with Moran Spectral Randomization

Description

This function decomposes the variation in an outcome variable into four fractions: a) the influence of covariates, b) joint influence of covariates and space, c) the influence of space, and d) unexplained residual variation. Moran spectral randomization is applied to obtain the expected value of the coefficient of determination adjusted for spurious correlations.

Usage

```r
vp(y, x = NULL, evecs = NULL, msr = 100)
```

Arguments

- `y`: outcome vector
- `x`: vector/ matrix of covariates
- `evecs`: selected eigenvectors
- `msr`: number of permutations to compute the expected value under H0

Value

Returns an object of class `vpart` which provides the following information:

- `R2`: unadjusted fractions of explained variation
- `adjR2`: adjusted fractions (based on Moran spectral randomization)
- `msr`: number of permutations to obtain the expected value under H0

Note

The adjusted R-squared values are obtained by: 1-(1-R2)/(1-E(R2|H0)). For fractions [ab] and [a], Moran spectral randomization is used to derive E(R2|H0). To this end, the rows in matrix (or column vector) `x` are randomly permuted in order to preserve the correlation structure (see e.g., Clappe et al. 2018).

See Also

- `lmFilter`, `getEVs`

Examples

```r
data(fakedata)
E <- getEVs(W=W,covars=NULL)$vectors
(VIF <- vif.ev(x=fakedataset$x1,evecs=E[,1:10]))
```
Author(s)

Sebastian Juhl

References


See Also

getEVs

Examples

data(fakedata)
E <- getEVs(W=W,covars=NULL)$vectors

(partition <- vp(y=fakedataset$x1,evecs=E[,1:10],msr=100))
Index

* datasets
  fakedata, 2
* dataset
  fakedata, 2

fakedata, 2
fakedataset (fakedata), 2

glmFilter, 4, 4, 10, 13–15

lmFilter, 4, 7, 8, 13–15, 18, 19

MI.decomp, 11, 17
MI.ev, 4, 12, 12, 15, 17
MI.local, 12, 14, 17
MI.local (MI.vec), 15
MI.resid, 7, 10, 12, 13, 17
MI.sf, 4, 12, 13, 14, 17
MI.vec, 12, 14, 15, 17

optim, 7

partialR2, 4, 17

summary.spfilter (lmFilter), 8

vif.ev, 4, 18
vp, 19

W (fakedata), 2