

spmoran (ver. 0.2.0): An R package for Moran eigenvector-based scalable spatial additive mixed modeling

Daisuke Murakami (e-mail: dmuraka@ism.ac.jp)

Department of Data Science, Institute of Statistical Mathematics, Tokyo, Japan

Another example using the Boston house price data and the sample code is available from

<https://github.com/dmuraka/spmoran/tree/master>

Table of Contents

1. Introduction	2
2. Moran eigenvector-based spatial regression models	4
2.1. Moran eigenvectors	4
2.2.1. What are Moran eigenvectors?	4
2.2.2. Specification of Moran eigenvectors	5
2.2. Spatial regression models	7
2.2.1. Eigenvector spatial filtering (ESF) model	7
2.2.2. Random effects ESF (RE-ESF) model	9
2.2.3. RE-ESF model for large samples	11
2.3. Spatially and non-spatially varying coefficient models	12
2.3.1. Varying coefficient modeling	12
2.3.2. Spatially varying coefficient model	14
2.3.3. Spatially and non-spatially varying coefficient model	18
2.4. Models with group effects	23
2.4.1. Multilevel model	23
2.4.2. Small area estimation	25
2.4.2. Longitudinal/panel data model	28
2.5. Spatially filtered unconditional quantile regression model	32
2.6. Spatial prediction	35
3. Low rank spatial econometric models	38
3.1. Spatial weight matrix and their eigenvectors	38

3.2. Spatial regression models	39
3.2.1. Low rank spatial lag model.....	39
3.2.2. Low rank spatial error model	41
4. Tips for modeling large samples	42
4.1. Eigen-decomposition	42
4.2. Parameter estimation.....	44
4.3. For very large samples (e.g., millions of samples).....	44
5. Future updates	45
6. Reference.....	45

1. Introduction

This package provides functions estimating Moran eigenvector-based scalable spatial additive mixed models and related spatial models. In concrete, this package implements standard spatial regression models and extensions, including spatially and non-spatially varying coefficient model, models with group effects, spatial unconditional quantile regression model, and low rank spatial econometric models. All these models are estimated computationally efficiently (see Table 1).

These models are extensions of the random effects eigenvector spatial filtering (RE-ESF) approach (Murakami and Griffith, 2015). Just like the classical ESF (see Griffith, 2003; Tiefelsdorf and Griffith, 2007), this approach efficiently eliminates residual spatial dependence using a spatial process that is interpretable in terms of the Moran coefficient (Moran I statistic).

Table 1: Summary of functions. The first six implements low rank GPs that are interpretable in terms of the Moran coefficient, and the last two are low rank spatial econometric models.

Function	Spatial dependence		Varying coefficients		Group effects	Quantile regress. (uncond.)	Computation		Function for ME extraction	Function for spatial prediction
	Residual	Explained variable	Spatial	Non-spatial			Fast	Memory free		
esf	■									predict0
resf				■	■			■	meigen or meigen_f	predict0
resf_vc			■							predict0_vc
resf_qr						■				
besf	■			■				■		predict0
besf_vc			■							predict0_vc
lsm		■							weigen	
lsem	■						■			

This vignette illustrates how to use functions in this package through land price analyses in the Ibaraki prefecture, Japan. Explained variables are logged land prices in 2015 (JPY/m²; sample size: 647; Figure 1). Explanatory variables are as listed in Table 2. All these variables are downloaded from the National Land Numerical Information download service (<http://nlftp.mlit.go.jp/ksj-e/index.html>).

Here is the first 5 rows in the dataset. “px” and “py” are spatial coordinates:

```
> data[1:5,]
      px      py ln_price  station  tokyo urban flood  city_name
1 19235.25 -4784.562 10.12663 4.0109290 43.38504 1 1.5 Tsukubamirai-shi
2 16450.37 -8782.851 10.83565 0.8977986 43.38504 1 0.0 Toride-shi
3 17673.30 -8351.802 10.63345 0.5596742 43.38504 1 0.0 Toride-shi
4 17824.50 -7704.343 9.87817 0.8504618 43.38504 0 0.0 Toride-shi
5 67334.31 58001.724 10.12262 3.1660661 140.95839 1 0.0 Hitachiota-shi
```

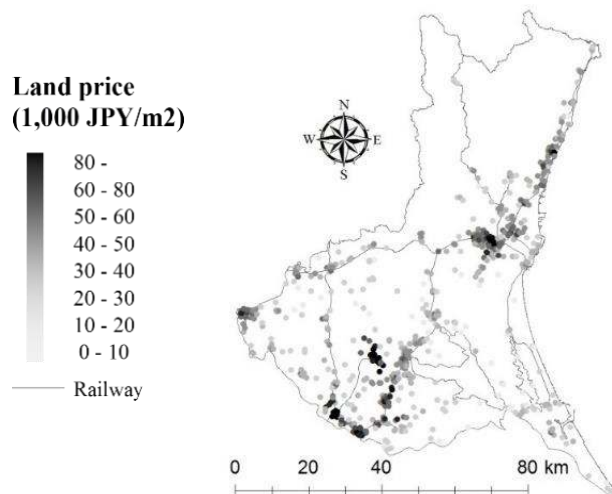


Figure 1. Officially assessed residential land prices in 2015 (right) in the Ibaraki prefecture/

Table 2. Explanatory variables

Variables	Description
tokyo	Logarithm of the distance from the nearest railway station to Tokyo Station [km]
station	Logarithm of the distance to the nearest railway station [km]
flood	Anticipated inundation depth [m]
urban	1 if the site is in an urban promotion land and 0 otherwise

2. Moran eigenvector-based spatial regression models

This package assumes the following analysis steps: (a) define Moran eigenvectors; (b) spatial regression using these eigenvectors. Hereafter, Section 2.1 explains (a) whereas Sections 2.2 to 2.5 explain (b).

2.1. Moran eigenvectors

2.2.1. What are Moran eigenvectors?

Moran eigenvectors (MEs) $\{\mathbf{e}_1, \dots, \mathbf{e}_L\}$ are the eigenvectors extracted from a doubly-centered spatial proximity matrix where \mathbf{e}_1 is the ME corresponding to the largest eigenvalue λ_1 , \mathbf{e}_2 is the ME corresponding to the second largest eigenvalue λ_2 , and so forth. Figure 2 plots the 1st, 5th, and 25th eigenvectors. As illustrated in this figure, MEs corresponding to larger eigenvalue describes larger-scale map pattern.

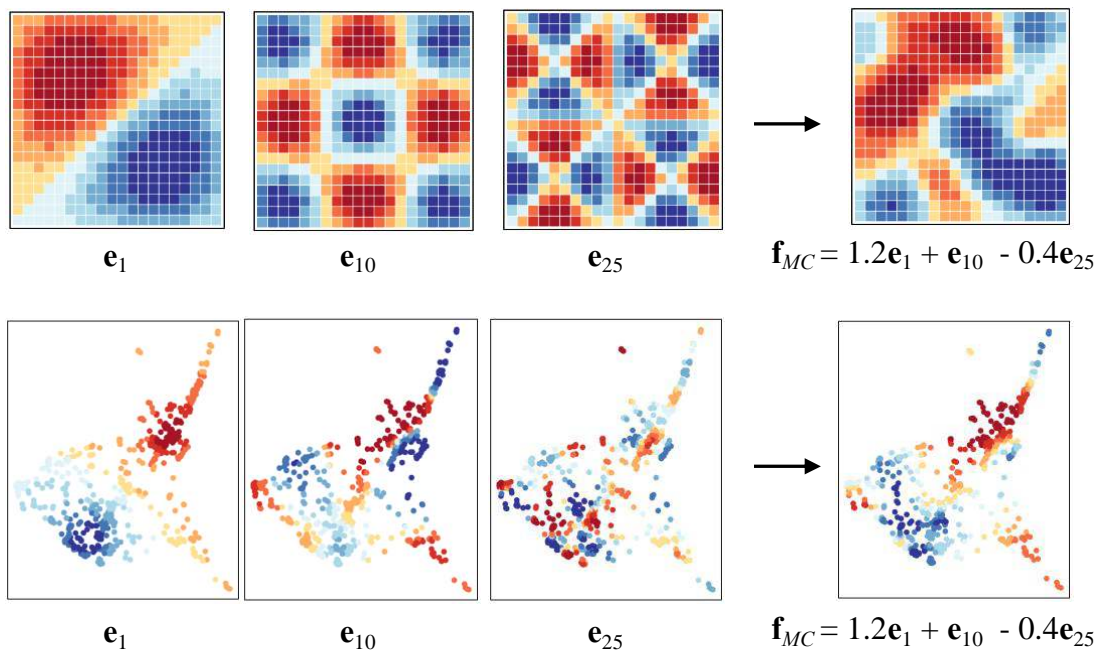


Figure 2. Examples of the MEs (\mathbf{e}_1 , \mathbf{e}_{10} , \mathbf{e}_{25}) and their weighted sum (\mathbf{f}_{MC}), which is used in regression analysis. Here, only three eigenvectors are considered just for illustration; all the MEs corresponding to positive eigenvalues are considered in this package by to accurately model positive spatial dependence.

MEs are interpretable in terms of the Moran coefficient (MC; or Moran's I), which is a spatial dependence diagnostic statistic indicating larger positive value in the presence of stronger positive spatial dependence. Specifically, the following relationship holds:

$$MC[\mathbf{e}_l] \propto \lambda_l, \quad (1)$$

where \propto represents proportion. Eq.(1) suggests that \mathbf{e}_1 , which had the largest-scale map pattern, achieves the largest MC value, \mathbf{e}_2 , which had the second largest-scale pattern, achieves the second largest MC value, and so on. Thus, MEs furnish distinct map pattern descriptions of latent spatial dependence, with each level being indexed by the MC value (Griffith, 2003).

In this package, MEs corresponding to positive eigenvalue are used to model positively dependent spatial process behind observations. Suppose that $e_l(s_i)$ is the i -th element of the l -th eigenvector \mathbf{e}_l , or the value of the eigenvector at site s_i . Then, the following model is used to model the spatial process at site s_i :

$$f_{MC}(s_i) = \sum_{l=1}^L e_l(s_i)\gamma_l, \quad (2)$$

where L is the number of positive eigenvalue and γ_l is a coefficient. Eq.(2) models the spatial process by estimating the coefficients. See Figure 2 for examples. ESF defines γ_l by a fixed coefficient whereas RE-ESF defines γ_l by a random coefficient. I prefer the latter because of the following reasons: (i) the random specification tends to be more accurate (e.g., Donegan et al., 2020); (ii) in the random case, $f_{MC}(s_i)$ yields an approximate Gaussian process, which is widely accepted for spatial process modeling in geostatistics (Murakami and Griffith, 2015). Because of these reasons, most functions in this package models $f_{MC}(s_i)$ while assuming γ_l as random coefficients.

2.2.2. Specification of Moran eigenvectors

To model positively dependent spatial process, the meigen function extracts the MEs corresponding to positive eigenvalue using the following command:

```
> coords <- data[,c("px", "py")]
> meig <- meigen(coords = coords)
```

meig includes extracted eigenvectors and eigenvalues; they are extracted from a matrix that doubly centers a spatial proximity matrix \mathbf{C} . The (i, j) -th element of \mathbf{C} is given by $\exp(-d_{i,j}/r)$, where $d_{i,j}$ is the Euclidean distance between sites i and j , and r is the longest distance in the minimum spanning tree covering the sample sites (Dray et al., 2006; Murakami and Griffith, 2015). The exponential kernel can be substituted by Gaussian kernel (`model = "gau"`) or spherical kernel (`model = "sph"`).

It is often the case that there are multiple samples in the same site. For example, spatial coordinates of condominium units are usually given by buildings. Panel data has multiple samples in

the same site/unit with different time points. For such data, an argument `s_id` in the `meigen` (or `meigen_f`) function is useful to define MEs by not individual samples but groups. For instance, if `s_id` is given by ID of condominiums buildings, MEs are defined by buildings based on the proximity among buildings. If spatial coordinates of samples within the same group are not identical, the geometric center of the group is used to extract the MEs.

The distance-based **C** can be replaced with other spatial connectivity matrix. For example, here is a code to prepare the 4-nearest-neighbor-based **C** matrix:

```
> library( spdep )
> col.knn <- knearneigh( coordinates( coords ), k = 4 )
> cmat <- nb2mat( knn2nb( col.knn ), style = "B" )
> meigB <- meigen( cmat = cmat )
```

If spatial connectivity matrix is not symmetric like the 4-nearest neighbor-based **C**, the `meigen` function symmetrizes it by taking $(\mathbf{C} + \mathbf{C}')/2$. In cases with binary connectivity-based **C** (e.g. proximity-based **C**; k -nearest-neighbor-based **C**), $\lambda_l/\lambda_1 > 0.25$ is a standard threshold¹. The threshold is implemented as follows:

```
> meigB <- meigen( cmat = cmat, threshold = 0.25 )
```

Other connectivity matrices including road distance-based matrix and economic connectivity-based matrix are available (although I recommend the default exponential function because of the stability).

While eigen-decomposition is computationally demanding in general, the `meigen_f` function performs fast approximation of MEs for large samples (see Murakami and Griffith, 2019a). The command for the fast ME extraction is as follows:

```
> meig_f <- meigen_f( coords = coords )
```

By default, the first 200 eigenvectors are approximated based on a simulation result in Murakami and Griffith (2019a). While `meigen` took 243.79 seconds for 5,000 samples, `meigen_f` took only 0.38 seconds (see Section 4.1 for further details). Currently, this approximation is available only for the exponential, Gaussian, and spherical kernels.

Sections 2.2 to 2.6 explains how to use the extracted MEs in spatial regression modeling.

¹ The threshold $\lambda_l/\lambda_1 > 0.25$ attempts to capture roughly 95% of the variations attributable to positive spatial dependence (Griffith and Chun, 2014).

2.2. Spatial regression models

2.2.1. Eigenvector spatial filtering (ESF) model

The classical ESF model is formulated as follows:

$$y_i = \sum_{k=1}^K x_{i,k} \beta_k + f_{MC(F)}(s_i) + \varepsilon_i, \quad \varepsilon_i \sim N(0, \sigma^2). \quad (3)$$

$f_{MC(F)}(s_i) = \sum_{l=1}^L e_l(s_i) \gamma_{l(F)}$ where $\gamma_{l(F)}$ is a fixed coefficient (see Eq.2). $f_{MC(F)}(s_i)$ captures residual spatial dependence to estimate and infer regression coefficients appropriately. If spatial dependence in residuals is ignored, coefficient standard error tends to be underestimated, and the statistical significance is overestimated. Consideration of spatial dependence is needed to avoid such problem.

The ESF model is estimated by the following steps: (i) MEs corresponding to positive eigenvalue are extracted using the meigen function; (ii) L MEs selected through a stepwise variable selection procedure; (iii) the ESF model Eq.(3) with the selected eigenvectors is estimated using the ordinary least squares method. Binary connectivity matrix (e.g., knn-based \mathbf{C}) is used for many ESF studies.

The following command estimates the linear ESF model:

```
> y      <- data[ , "ln_price" ]                # Explained variables
> x      <- data[ , c( "station", "tokyo", "city", "flood " ) ] # Explanatory variables
> meig   <- meigB                              #Moran's eigenvectors (knn-based C)
> e_res  <- esf( y = y, x = x, meig = meig, vif = 10, fn = "r2" )
```

To cope with multicollinearity, eigenvectors can be selected so that the variance inflation factor (VIF), which is an indicator of multicollinearity, does not to exceed 10. It is implemented by specifying `vif = 10`. The stepwise eigenvector selection is performed through an adjusted R^2 maximization (`fn = "r2"`; default), Akaike information criterion (AIC) minimization (`fn = "aic"`), or Bayesian Information criterion (BIC) minimization (`fn = "bic"`). Alternatively, if `fn = "all"`, all the MEs corresponding to positive eigenvalue are considered.

Summary of the estimation result is displayed as shown in the next page. The “Coefficients” section describes estimated coefficients and their statistical significance. Station (-) and city (+) are statistically significant at the 0.1% level. These results confirm higher land price nearby railway station areas and other urban areas. Flood is positively significant at the 10% level. This result suggests that influence from flood disaster, which is expected to be negative, is inappropriately reflected to land price. The “Spatial effects (residuals)” section describes the standard error (SE) (`SE`) and the scaled MC (`Moran.I/max(Moran.I)`) of the estimated spatial process $f_{MC(F)}(s_i)$. The SE is compatible to the

residual SE (`resid_SE`). The result suggests the presence of residual spatial dependence. The scaled MC takes a value between 0 and 1 in which 0 suggests the smallest-scale spatial dependence whereas 1 means the largest-scale spatial dependence. The estimated value of 0.137 suggests that small-scale residual spatial dependence is estimated. Finally, the “Error statistics” section summarizes error statistics, including residual standard error (`resid_SE`), adjusted R^2 (`adjR2`), log-likelihood (`logLik`), AIC, and BIC:

```
> e_res
Call:
esf(y = y, x = x, vif = 10, meig = meig, fn = "r2")

----Coefficients-----
              Estimate          SE      t_value      p_value
(Intercept)  9.932080e+00  0.0587240255  169.13146372  0.000000e+00
station      -6.911515e-02  0.0065601988  -10.53552610  5.070594e-24
tokyo        -2.846888e-05  0.0004214075   -0.06755664  9.461599e-01
city         6.738630e-01  0.0360500253   18.69244166  2.121536e-62
flood        2.795299e-02  0.0142681894    1.95911280  5.053884e-02

----Spatial effects (residuals)-----
              Estimate
SE              0.3163393
Moran.I/max(Moran.I) 0.1372278

----Error statistics-----
              stat
resid_SE      0.3542671
adjR2         0.6987400
logLik       -239.0702859
AIC           510.1405718
BIC           581.6981125
```

VIF values are displayed by the following command:

```
> e_res$vif

              VIF
station      1.367917
tokyo        1.225594
city         1.282930
flood        1.208189
sf4          1.167728
sf9          1.017697
sf12         1.142611
sf31         1.084662
sf33         1.032077
```


sf45	1.035118
sf32	1.095973
sf26	1.012234
sf6	1.059948
sf20	1.016059

While binary spatial proximity matrix is frequently used in regional science, distance-based matrix is popular in ecological studies. The distance-based ESF is implemented as follows:

```
> meig <- meigen( coords=coords ) #Moran's eigenvectors (distance-based C)
> e_res <- esf( y=y, x=x, meig=meig, fn="r2")
```

The distance-based ESF is often called Moran eigenvector maps (MEMs) or a principal coordinate neighborhood matrix (PCNM) (see Legendre and Legendre, 2012).

A major disadvantage of ESF is the computational cost. To overcome the limitation, Murakami and Griffith (2019a) proposed a fast ESF, which is implemented as

```
> meig_f <- meigen_f( coords = coords )
> e_res <- esf( y = y, x = x, meig = meig_f, fn = "all" )
```

Here, all the eigenvectors in `meig_f` are used without selecting them by specifying `fn = "all"`. It is acceptable for medium to large samples).

2.2.2. Random effects ESF (RE-ESF) model

The RE-ESF model is formulated as follows:

$$y_i = \sum_{k=1}^K x_{i,k} \beta_k + f_{MC}(s_i) + \varepsilon_i, \quad \varepsilon_i \sim N(0, \sigma^2), \quad (4)$$

As with the classical ESF, this model is useful to estimate and infer regression coefficients in the presence of residual spatial dependence. Unlike ESF, $f_{MC}(s_i)$ is given by a random spatial process approximating a Gaussian process (GP) that is frequently used for spatial process modeling in geostatistics (see Section 2.1). In other words, the RE-ESF model is an approximate GP interpretable in terms of the MC.

The RE-ESF model is estimated by the following steps: (i) $L (< N)$ MEs are extracted; (ii) parameters are estimated by the Type II maximum likelihood (ML) method or the restricted maximum likelihood (REML) method. I recommend REML because of a theoretical result for semiparametric

linear model including Eq.(4) that REML tends not to have local optima, and stable (Reiss and Ogden, 2011).

The REML estimation is implemented by the following command:

```
> meig <- meigen( coords = coords ) #Moran's eigenvectors (distance-based C)
> r_res <- resf( y = y, x = x, meig = meig )
```

The REML is replaced with ML by specifying `method = "ml"` in the `resf` function. Estimation result is displayed as below:

```
> r_res
Call:
resf(y = y, x = x, meig = meig)

----Coefficients-----
              Estimate      SE    t_value    p_value
(Intercept)  9.9902450636  0.169898966  58.8010941  0.000000e+00
station      -0.0792873434  0.009597978  -8.2608384  8.881784e-16
tokyo        -0.0003707685  0.001796579  -0.2063747  8.365681e-01
city         0.6857654664  0.036926315  18.5711862  0.000000e+00
flood        -0.0043678968  0.014784086  -0.2954459  7.677553e-01

----Variance parameter-----

Spatial effects (residuals):
              (Intercept)
random_SE      0.4336549
Moran.I/max(Moran.I)  0.1302975

----Error statistics-----
              stat
resid_SE      0.3116842
adjR2(cond)   0.7653482
rlogLik       -262.9627094
AIC           541.9254189
BIC           577.7041892
```

As same as ESF, station (-) and city (+) are statistically significant while tokyo is not. In contrast, unlike ESF, flood is insignificant. Because RE-ESF tends to outperform ESF in terms of the estimation accuracy of regression coefficients and their standard errors, the RE-ESF result might be more reliable. The “Variance parameter” section summarizes estimated variance parameters. SE of the estimated spatial process ($f_{MC}(s_i)$) (`random_SE`) is greater than the residual SE (`resid_SE`). The scaled MC (`Moran.I.max(Moran.I)`) is as small as ESF. Thus, strong small-scale spatially variation is estimated.

The “Error statistics” section displays residual SE (`resid_SE`), the adjusted conditional R^2 (`adjR2(cond)`), the restricted log-likelihood (`rlogLik`), `AIC`, and `BIC`. `rlogLik` is replaced with `loglik`,

which is the log-likelihood, if `method = "ml"`. Note that, when REML is used, AIC and BIC are comparable only with models with the same explanatory variables.

2.2.3. RE-ESF model for large samples

Although the `resf` function implements the fast ML/REML of Murakami and Griffith (2019b) that is available for large samples, the `meigen` function, which performs the usual eigen-decomposition, is computationally demanding, and not available for large samples (e.g., 10 thousand samples). To avoid the problem, the `meigen` function may be replaced with the `meigen_f` function as follows:

```
> meig_f <- meigen_f( coords = coords )  
> r_res2 <- resf( y = y, x = x, meig = meig_f )
```

While `meigen` took 243.79 seconds for 5,000 samples, `meigen_f` took only 0.38 seconds (see Section 4.1 for further details). Still, the `meigen_f` function stores $L \times N$ eigenvector matrix ($L = 200$ by default). In other words, this function consumes a considerable amount of memories, and not available for very large samples such as millions of samples. To break the bottleneck, the `besf` function implements a memory-free procedure for the ME approximation and RE-ESF modeling (Murakami and Griffith, 2019c). The command is as follows:

```
> r_res3 <- besf( y = y, x = x, coords = coords)
```

This function implements a parallel computation by default. So, faster than the `resf` function. We confirm that the `besf` function is available for 10 million samples. See Section 4 for further detail for modeling large samples.

2.3. Spatially and non-spatially varying coefficient models

2.3.1. Varying coefficient modeling

Effects from covariates can vary depending on covariate value. For example, distance to railway station might have strong impact if the distance is small while weak if the distance is large. To capture such effect, the `resf` function estimates coefficients varying with respect to the covariate value, which I call non-spatially varying coefficients (NVCs). If `nvc = TRUE`, the `resf` function estimates the following model considering residual spatial dependence ($f_{MC}(s_i)$) and NVCs ($\beta_{i,k}$):

$$y_i = \sum_{k=1}^K x_{i,k} \beta_{i,k} + f_{MC}(s_i) + \varepsilon_i, \quad \beta_{i,k} = b_k + f(x_{i,k}), \quad \varepsilon_i \sim N(0, \sigma^2), \quad (5)$$

where $f(x_{i,k})$ is the function of the covariate $x_{i,k}$. The $f(x_{i,k})$ function is given by the natural spline generated from the covariate. The command estimating the model and the output are as follows:

```
> r_res2<-resf(y=y, x=x, meig=meig,nvc=TRUE)
> r_res2

----Non-spatially varying coefficients (summary)----

Coefficients:
  station      tokyo      city      flood
Min.   :-0.19235  Min.   :-0.001083  Min.   :0.6344  Min.   :-0.002225
1st Qu.:-0.18242  1st Qu.:-0.001083  1st Qu.:0.6344  1st Qu.:-0.002225
Median :-0.16061  Median :-0.001083  Median :0.6344  Median :-0.002225
Mean   :-0.15259  Mean   :-0.001083  Mean   :0.6344  Mean   :-0.002225
3rd Qu.:-0.13782  3rd Qu.:-0.001083  3rd Qu.:0.6344  3rd Qu.:-0.002225
Max.   :-0.05409  Max.   :-0.001083  Max.   :0.6344  Max.   :-0.002225

Statistical significance:
              station tokyo city flood
Not significant      0  647  0  647
Significant (10% level)  0  0  0  0
Significant ( 5% level)  0  0  0  0
Significant ( 1% level) 647  0 647  0

----Variance parameter-----

Spatial effects (residuals):
              (Intercept)
random_SE      0.4141524
Moran.I/max(Moran.I) 0.1303574

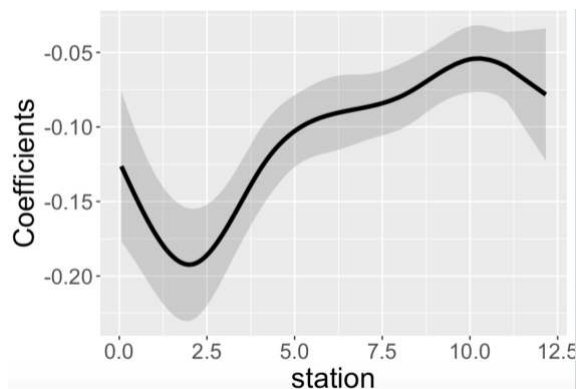
Non-spatially varying coefficients:
              station tokyo city flood
random_SE 0.01004439  0  0  0

----Error statistics-----
              stat
resid_SE      0.2969830
adjR2(cond)   0.7866274
rlogLik      -241.9254246
AIC           501.8508493
BIC           542.1019659
```

By default, the `resf` function selects coefficient type (constant or NVC) through a BIC minimization, which can be replaced with an AIC minimization by specifying `penalty = "aic"`. Based on the result, the coefficient on station is estimated non-spatially varying while the coefficients on tokyo, city, and flood are estimated constant. The [Variance parameter](#) section summarizes the standard errors of the estimated spatial process ($f_{MC}(s_i)$) and the NVCs, together with the scaled MC value (Moran.I/max(Moran.I)) of $f_{MC}(s_i)$. Note that all the coefficients can be given by NVCs by additionally specifying `nvc_sel = FALSE` in the `resf` function.

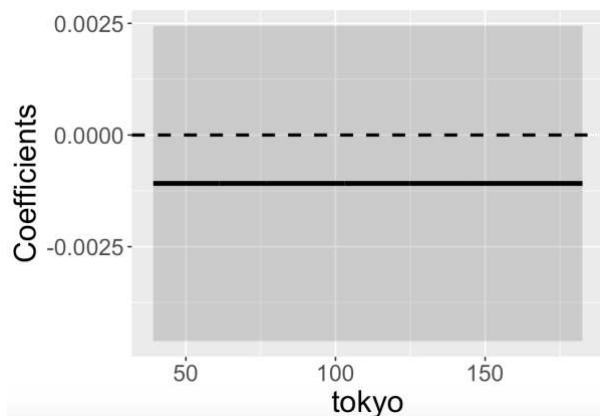
The `plot_n` function is available to plot the estimated NVCs. An argument `xnum` specifies the number of NVC. For instance, if `xnum = 1`, the estimated NVC on the column of x (station) is plotted as below. The line denotes the NVC estimates and the grey area represents the 95 % confidence interval. This plot suggests that station has stronger negative impact nearby railway station areas.

```
> plot_n(r_res2, xnum = 1)
```



For coefficients that are estimated constant, the constant estimates are plotted together with the 95 % confidential interval. The dotted line means zero.

```
> plot_n(r_res2, xnum = 2)
```



2.3.2. Spatially varying coefficient model

Murakami et al. (2017) and Murakami and Griffith (2019b) showed that a ME-based SVC (M-SVC) modeling approach outperforms the geographically weighted regression (GWR) approach, which is a standard SVC modeling approach, in terms of coefficient estimation accuracy and computational time. The SVC model is formulated as

$$y_i = \sum_{k=1}^K x_{i,k} \beta_{i,k} + f_{MC}(s_i) + \varepsilon_i, \quad \beta_{i,k} = b_k + f_{MC,k}(s_i), \quad \varepsilon_i \sim N(0, \sigma^2), \quad (6)$$

The k -th SVCs is defined as $\beta_{i,k} = [\text{constant mean } b_k] + [\text{spatially varying component } f_{MC,k}(s_i)]$. Advantages of the M-SVC modeling approach relative to GWR are as follows.

- (1) The M-SVC model estimates spatial scale (or the MC value) of individual SVC whereas the classical GWR assumes a common scale across SVCs. Such coefficient-wise scale estimation was implemented in the multiscale GWR (MGWR; e.g., Fotheringham et al., 2017).
- (2) The k -th SVC can be replaced with a constant coefficient by assuming $\beta_{i,k} = b_k$. In other words, our model allows for assuming SVCs on some covariates while constant coefficients on the others. Semiparametric GWR (SGWR) employs such an assumption (Nakaya et al., 2009).
- (3) This model is fast and free from memory limitation. Thus, it is available for 10 million samples (see Murakami and Griffith, 2019c).
- (4) Unlike GWR models, the M-SVC model is easily extended without losing its computational efficiency as I will explain later.

For illustration, I assume SVCs on tokyo, station, city, and flood whereas constant coefficients on px and py (spatial coordinates). The command for the M-SVC modeling is as follows:

```
> xv      <- x[, c("tokyo", "station", "city", "flood" )] #x with spatially varying coefficients
> xconst  <- x[, c("px", "py" )]                       #x with constant coefficients
> meig    <- meigen( coords = coords )                 #Moran's eigenvectors (distance-based C)
> rv_res  <- resf_vc( y = y, x = xv, xconst = xconst, meig = meig )
```

Remember that SVC models tend to suffer from multicollinearity (Wheeler and Tiefelsdorf, 2005). To address the problem, the `resf_vc` function selects coefficients type (constant or SVC) through a Bayesian information criterion (BIC) minimization by default. The selection can be replaced with an Akaike information criterion (AIC) minimization-based selection by specifying `penalty = "aic"` in the `resf_vc` function. Alternatively, if `x_sel = FALSE` is added, SVCs are assumed on all the covariates (see Section 2.3.3 for further detail about model selection).

The estimation result with the BIC-based model selection is as follows:

```

> rv_res
Call:
resf_vc(y = y, x = x, meig = meig)

----Spatially varying coefficients on x (summary)----

Coefficient estimates:
(Intercept)      station      tokyo      city      flood
Min.   : 9.155   Min.   :-0.20724   Min.   :-0.003905   Min.   :-0.0823   Min.   :-0.004508
1st Qu.:10.001   1st Qu.: -0.15210   1st Qu.: -0.003905   1st Qu.: 0.5726   1st Qu.: -0.004508
Median :10.195   Median : -0.12128   Median : -0.003905   Median : 0.6953   Median : -0.004508
Mean   :10.196   Mean   : -0.11578   Mean   : -0.003905   Mean   : 0.6887   Mean   : -0.004508
3rd Qu.:10.335   3rd Qu.: -0.07890   3rd Qu.: -0.003905   3rd Qu.: 0.8411   3rd Qu.: -0.004508
Max.   :10.995   Max.   : 0.04851   Max.   : -0.003905   Max.   : 1.1268   Max.   : -0.004508

Statistical significance:
                Intercept station tokyo city flood
Not significant      0      115    647  29   647
Significant (10% level)  0      64     0   12     0
Significant ( 5% level)  0      90     0   30     0
Significant ( 1% level) 647    378     0  576     0

----Constant coefficients on xconst-----
                Estimate      SE      t_value      p_value
px -6.113010e-07 4.023963e-06 -0.1519152 0.8793085
py 6.084577e-06 3.963662e-06 1.5350898 0.1253255

----Variance parameters-----

Spatial variation (coefficients on x):
                (Intercept) station tokyo      city flood
random_SE      0.3878430 0.07940766      0 0.32763842      0
Moran.I/max(Moran.I) 0.0886329 0.04648084      NA 0.07590154      NA

----Error statistics-----
                stat
resid_SE      0.2669825
adjR2(cond)   0.8267360
rlogLik      -253.2034445
AIC          534.4068890
BIC          597.0197371

```

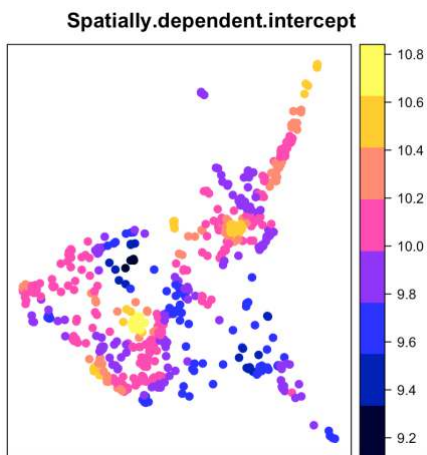
The [Spatially varying coefficients on x \(summary\)](#) section summarizes the estimated SVCs and their statistical significance. Based on the result, coefficients on station and city (and intercept) are estimated spatially varying whereas those on tokyo and flood are estimated constant. SVCs on station are statistically significant at the 1 % level at 378 sample sites whereas SVCs on city are significant at the same level at 576 sites. Coefficients on tokyo and flood are not significant. The [Constant coefficients](#) section summarizes estimated constant coefficients on `xconst`, including `px` and `py`.

The [Variance parameters](#) section summarizes parameters determining map patterns of the SVCs. “`random_SE`” represents standard error of the estimated SVCs. Because coefficients on tokyo and flood are estimated constant, their standard errors are zeros zero. `Moran.I/max(Moran.I)` denotes

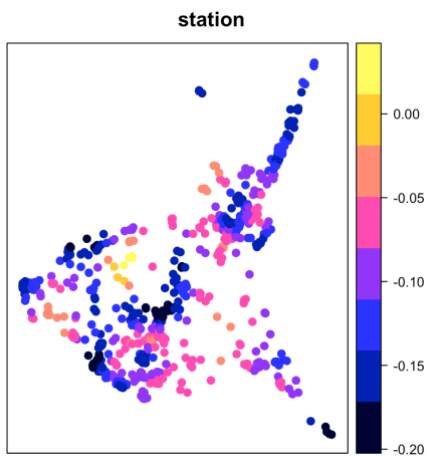
the scaled MC value of the map pattern. The values suggest that each SVC has relatively local map patterns. Lastly, the [Error statistics](#) section summarizes model errors.

The `plot_s` function is available to map the estimated coefficients. An argument `xnum` specifies an SVC where `xnum = 0` means spatially varying intercept, and `xnum > 0` means the column number of `x` (except for intercept). Commands are as follows:

```
> plot_s(rv_res, xnum=0) # Spatially dependent intercept
```

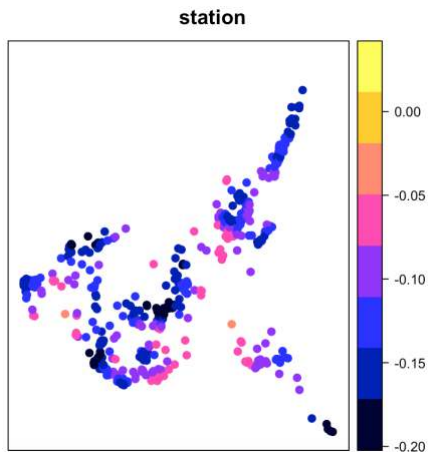


```
> plot_s(rv_res, xnum=1) # SVCs on station (the 1st column of x)
```



To plot only statistically significant estimates, an argument `pmax` specifies the maximum p-value. For example, The SVCs on station that are statistically significant at the 5 % level are plotted as

```
> plot_s(rv_res, xnum=1, pmax = 0.05 )
```



This result suggests that station has positive impact along major railway that though from the southwest to the northeast of the study area. For coefficients that are estimated constant, this function returns the following message without plotting.

```
> plot_s(rv_res, xnum=2)
```

Note: Coefficients are not plotted because they are constant

There are some arguments to specify color coding, dot size and so on. This function is useful to quickly check the result. For better visualization, I suggest using leaflet, mapview, or other R packages, or QGIS, ArcGIS, or other GIS software.

2.3.3. Spatially and non-spatially varying coefficient model

Coefficients can vary both spatially and non-spatially. Given that, Murakami and Griffith (2020) developed a spatially and non-spatially varying coefficient (SNVC) model which is defined as

$$y_i = \sum_{k=1}^K x_{i,k} \beta_{i,k} + f_{MC}(s_i) + \varepsilon_i, \quad \beta_{i,k} = b_k + f_{MC,k}(s_i) + f(x_{i,k}), \quad \varepsilon_i \sim N(0, \sigma^2). \quad (7)$$

This model defines the k -th regression coefficient $\beta_{i,k}$ at i -th site by [constant: b_k] + [SVC: $f_{MC,k}(s_i)$] + [NVC: $f(x_{i,k})$]. They showed that this model is more stable than SVC models, and does not suffer from spurious correlations among SVCs even if existing SVC models suffer from severe spurious correlations. Based on the result, they recommend using the SNVC model even when the analysis objective is estimating SVCs.

The coefficient $\beta_{i,k}$ includes the following specifications:

- Constant : $\beta_{i,k} = b_k$
- NVC : $\beta_{i,k} = b_k + f(x_{i,k})$
- SVC : $\beta_{i,k} = b_k + f_{MC,k}(s_i)$
- SNVC : $\beta_{i,k} = b_k + f_{MC,k}(s_i) + f(x_{i,k})$

The `resf_vc` function specify each coefficient as follows (see also Table 2).

SVCs are assumed on `x` by default. If `x_nvc = TRUE` is added, SNVCs are assumed on `x`. If `x_sel = TRUE` is added, SVC is included only if it improves the BIC value. Likewise, if `x_nvc_sel = TRUE` (and `x_nvz = TRUE`) is added, NVC is included if only it improves the BIC. Alternatively, `x_sel` may be specified by column numbers of `x`. In that case, SVCs are assumed only on the covariates in the corresponding columns. Similarly, `x_nvc_sel` can be given by column numbers assuming NVCs.

Coefficients on `xconst` are assumed constant by default. NVCs are assumed by adding `xconst_nvc = TRUE`. In addition, if `xconst_nvc_sel = TRUE` (default), NVC is included only when it improves the BIC. If `xconst_nvc_sel = FALSE`, NVCs are assumed across the coefficients. Alternatively, if `xconst_nvc_sel` is given by column numbers of `xconst`, NVCs are assumed only on the covariates in the corresponding columns.

Table 2: Arguments specifying the coefficients in the `resf_vc` function. TRUE is the default for `x_sel`, `x_nvc_sel`, and `xconst_nvc_sel` whereas FALSE is the default for `x_nvc` and `xconst_nvc`.

Covariates	Coefficients	Select SVC or Constant	Consider NVC	Select NVC or Constant
<code>x</code>	With SVC	<code>x_sel</code>	<code>x_nvc</code>	<code>x_nvc_sel</code>
<code>xconst</code>	Without SVC		<code>xconst_nvc</code>	<code>xconst_nvc_sel</code>

Here is a code estimating the full model with coefficients type selection.

```
> rv_res2<-resf_vc(y=y, x=x, xconst=xconst, meig=meig, x_nvc=TRUE, xconst_nvc=TRUE)
```

```
> rv_res2
```

```
----Spatially and non-spatially varying coefficients on x (summary)----
```

```
Coefficient estimates:
```

(Intercept)	station	tokyo	city	flood
Min. : 9.741	Min. :-0.28352	Min. :-0.010871	Min. :-0.1783	Min. :-0.002216
1st Qu.:10.552	1st Qu.: -0.20892	1st Qu.: -0.009988	1st Qu.: 0.5418	1st Qu.: -0.002216
Median :10.699	Median :-0.17912	Median :-0.007346	Median : 0.6818	Median :-0.002216
Mean :10.687	Mean :-0.16907	Mean :-0.007397	Mean : 0.6689	Mean :-0.002216
3rd Qu.:10.818	3rd Qu.: -0.13567	3rd Qu.: -0.004963	3rd Qu.: 0.8340	3rd Qu.: -0.002216
Max. :11.560	Max. : 0.02906	Max. :-0.003764	Max. : 1.2200	Max. :-0.002216

```
Statistical significance:
```

	Intercept	station	tokyo	city	flood
Not significant	0	20	308	36	647
Significant (10% level)	0	18	86	11	0
Significant (5% level)	0	56	253	36	0
Significant (1% level)	647	553	0	564	0

```
----Non-spatially varying coefficients on xconst (summary)----
```

```
Coefficient estimates:
```

px	py
Min. :-3.837e-06	Min. :5.743e-06
1st Qu.: -3.837e-06	1st Qu.:5.743e-06
Median :-3.837e-06	Median :5.743e-06
Mean :-3.837e-06	Mean :5.743e-06
3rd Qu.: -3.837e-06	3rd Qu.:5.743e-06
Max. :-3.837e-06	Max. :5.743e-06

```
Statistical significance:
```

	px	py
Not significant	647	647
Significant (10% level)	0	0
Significant (5% level)	0	0
Significant (1% level)	0	0

```
----Variance parameters-----
```

```
Spatial variation (coefficients on x):
```

	(Intercept)	station	tokyo	city	flood
random_SE	0.36862667	0.06623723	0	0.34409123	0
Moran.I/max(Moran.I)	0.06370044	0.05553843	NA	0.08895847	NA

```
Non-spatial variation (coefficients on x):
```

	station	tokyo	city	flood
random_SE	0.01236218	0.000844719	0	0

```
Non-spatial variation (coefficients on xconst):
```

	px	py
random_SE	0	0

```

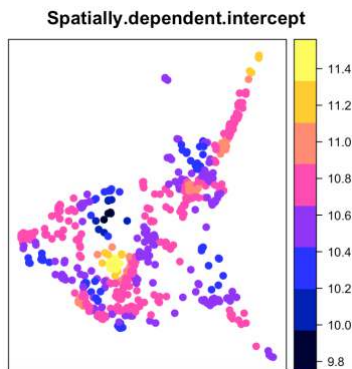
----Error statistics-----
                                stat
resid_SE      0.2537544
adjR2(cond)   0.8429830
rlogLik       -233.7105992
AIC           499.4211984
BIC           570.9787391

```

The BIC minimization result estimates SNVC on station, SVC on city (and intercept), NVC on tokyo, and constants on the others (see [Variance parameters](#) section).

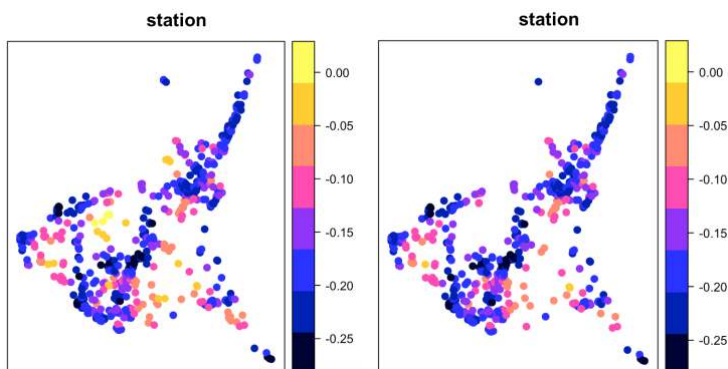
Estimated coefficients on x (SVC, NVC, or SNVC) are mapped using the `plot_s` function as below (see Section 2.3.2). Roughly speaking, coefficients on station suggests strong impact of station along principal railway, coefficients on tokyo suggests strong influence from Tokyo in the southwest region (Tokyo is located about 30km southwest from this area), and the coefficients on city suggests higher impact of urban area nearby major cities (there are central cities in each yellow areas). These results are intuitively reasonable.

```
> plot_s(rv_res2, xnum = 0)
```



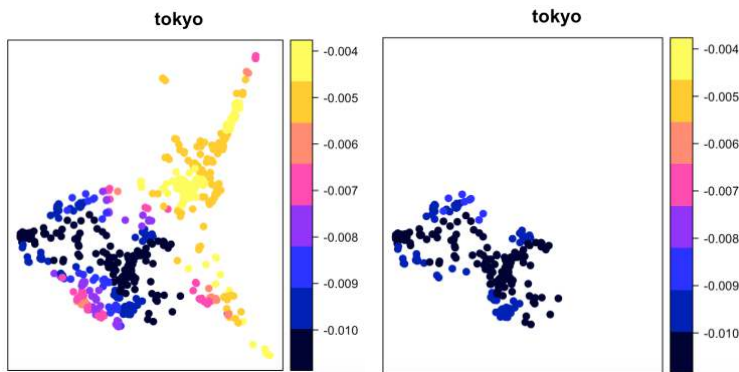
```
> plot_s(rv_res2, xnum = 1)
```

```
> plot_s(rv_res2, xnum = 1, pmax = 0.05) # pmax is the maximum p-value
```



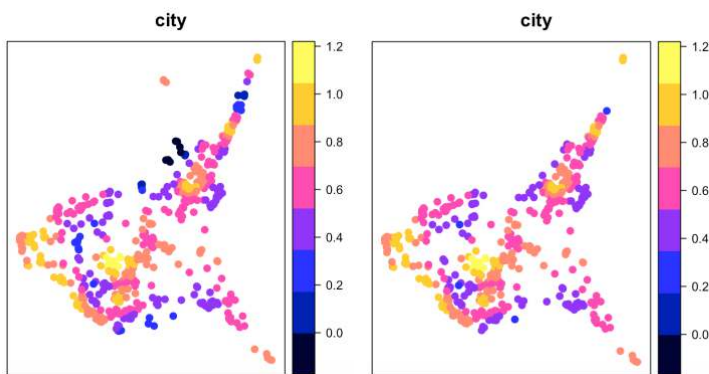
```
> plot_s(rv_res2, xnum = 2 )
```

```
> plot_s(rv_res2, xnum = 2, pmax = 0.05)
```



```
> plot_s(rv_res2, xnum = 3 )
```

```
> plot_s(rv_res2, xnum = 3, pmax = 0.05)
```

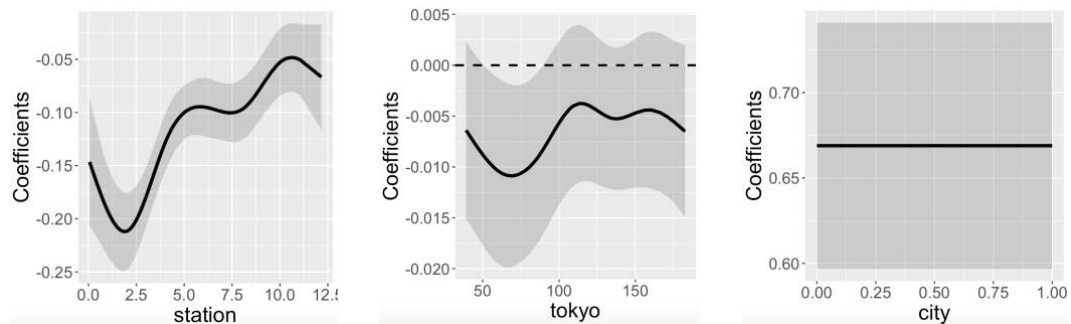


The `plot_n` function is available to plot the estimated NVCs on `x` or `xconst`. By default, it plots the NVC estimates on `x` together with their 95 % confidence intervals as follows:

```
> plot_n(rv_res2, xnum = 1 )
```

```
> plot_n(rv_res2, xnum = 2 )
```

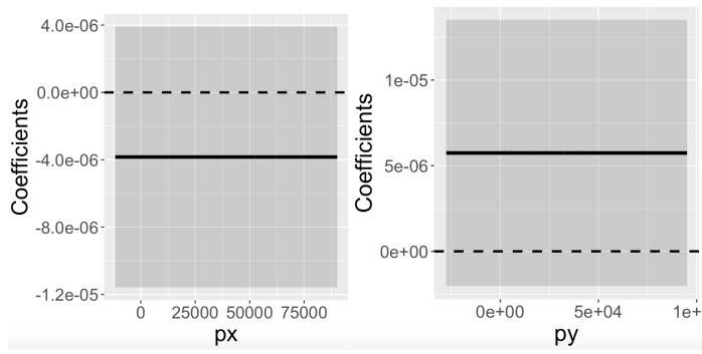
```
> plot_n(rv_res2, xnum = 3 )
```



If `xtype="xconst"` is added, NVCs on `xconst` is plotted.

```
> plot_n(rv_res2, xnum = 1, xtype="xconst" )
```

```
> plot_n(rv_res2, xnum = 2, xtype="xconst" )
```



2.4. Models with group effects

Two group effects are available in this package:

- (i) Spatially dependent group effects. Spatial dependence among groups are modeled instead of modeling spatial dependence among individual samples.
- (ii) Spatially independent group effects assuming independence across groups (usual group effects).

They are estimated in the `resf` and `resf_vc` functions. When considering (i) and (ii), the `resf` function estimates the following model (no NVC is assumed here):

$$y_i = \sum_{k=1}^K x_i \beta_k + f_{MC}(g_{I(0)}) + \sum_{h=1}^H \gamma(g_{I(h)}) + \varepsilon_i, \quad \varepsilon_i \sim N(0, \sigma^2), \quad (8)$$

where $g_{I(0)}, g_{I(1)}, \dots, g_{I(H)}$ represent group variables. $f_{MC}(g_{I(0)})$ represents (i) the spatially dependent group effects whereas $\gamma(g_{I(h)})$ represents (ii) the spatially independent group effects for the h -th group variable. The `resf_vc` function can estimate the following model with (i) and (ii) (again, no NVC is assumed):

$$y_i = \sum_{k=1}^K x_{i,k} \beta_{i,k} + f_{MC}(g_{I(0)}) + \sum_{h=1}^H \gamma(g_{I(h)}) + \varepsilon_i, \quad \varepsilon_i \sim N(0, \sigma^2). \quad (9)$$

$$\beta_{i,k} = b_k + f_{MC,k}(g_{I(0)}),$$

As examples, multilevel modeling, small area estimation, and panel data analysis are demonstrated below.

2.4.1. Multilevel model

Data often has multilevel structure. For example, school achievement of individual student changes depending on class and school. Condominium unit price depends not only on unit attributes but also building attributes. Ignorance of such multilevel structure can lead erroneous inference (typically, overestimation of statistical significance). Multilevel modeling is required to explicitly consider hierarchical structure behind data and evaluate statistical significance appropriately (see Snijders and Bosker, 2011).

The `resf` and `resf_vc` functions are available for multilevel modeling. An argument `x_group` specifies the group ID used to specify (ii) the spatially independent (i.e., usual) group effects. When considering one group variable, `x_group` is given by a vector of group ID. When considering H group variables, it is given by a matrix with its h -th column being the h -th group IDs. Here is a code estimating a land price model considering individual-level residual spatial dependence and (ii) municipality-level spatially independent group effects:

```
> xgroup <- data[, "city_name" ]      # Municipality name
> m1_res <- resf( y = y, x = x, meig = meig, xgroup = xgroup )
```

It is also possible to assume (i) group-level spatial dependence. This assumption is suitable, for example, when modeling condominium unit price whose spatial dependence arise among buildings rather than among units. (i) Spatially dependent group effects are introduced by specifying `s_id` in the `meigen` or `meigen_f` function using a group ID. Here is a code simultaneously estimating (i) spatially dependent group effects and (ii) spatially independent group effects:

```
> meig_g <- meigen( coords = coords, s_id = s_id )
> m2_res <- resf( y = y, x = x, meig = meig_g, xgroup = xgroup )
```

The BIC values of the two multilevel models are 535.8 and 652.7 (`m1_res$e` and `m2_res$e`). Relative to the without group effects model estimated in Section 2.2.2, the BIC value is improved especially in the first model assuming individual-level spatially dependence and spatially independent group effects. The estimates are displayed as below.

```
> m1_res
Call:
resf(y = y, x = x, xgroup = xgroup, meig = meig)

----Coefficients-----
              Estimate          SE    t_value  p_value
(Intercept) 10.252976269  0.213878697  47.9382772 0.0000000
station     -0.095555958  0.008230725 -11.6096649 0.0000000
tokyo       -0.003474515  0.002176735  -1.5962053 0.1109658
urban        0.764757108  0.038886177  19.6665541 0.0000000
flood       -0.011892280  0.014964623  -0.7946929 0.4271042

----Variance parameter-----

Spatial effects (residuals):
              (Intercept)
random_SE      0.4281196
Moran.I/max(Moran.I) 0.7898730

Group effects:
              xgroup
random_SE 0.3253219

----Error statistics-----
              stat
resid_SE      0.333405
adjR2(cond)   0.731082
rlogLik     -297.205044
AIC          612.410088
BIC          652.661204
```


This result shows that standard errors (`random_SE`) of (i) the spatial effects, (ii) group effects, and noise are 0.428, 0.325, and 0.333, respectively. If (ii) `xgroup` is ignored (`xgroup=NULL`), the BIC value becomes 835.0, which is worse than the presented result. It is suggested that land price has municipality-level variation.

The estimated (ii) group effects are displayed as follows:

```
> r_res$b_g
[[1]]
              Estimate      SE      t_value
xgroup_Ami-machi      0.054442162 0.13119263  0.41497880
xgroup_Bando-shi      0.131628850 0.14513678  0.90692967
xgroup_Chikusei-shi   -0.266329978 0.13922504 -1.91294590
xgroup_Daigo-machi    -0.033811902 0.20085140 -0.16834287
xgroup_Goka-machi     -0.197816323 0.19711741 -1.00354566
.
.
.
xgroup_Tsukuba-shi     0.533528999 0.10817352  4.93215894
xgroup_Tsukubamirai-shi -0.131326082 0.14140017 -0.92875474
xgroup_Ushiku-shi      0.294976331 0.12645680  2.33262527
xgroup_Yachiyo-machi  0.132015118 0.19394536  0.68068203
xgroup_Yuki-shi       -0.055251498 0.15485152 -0.35680307
```

2.4.2. Small area estimation

Small area estimation (SAE; Ghosh and Rao, 1994) is a statistical technique estimating parameters for small areas such as districts and municipality. SAE is useful to obtain reliable small area statistics from noisy data. Suppose that the raw data y_I in the I -th small area is defined by the mean of the N_I individual sample values $\{y_{1,I}, \dots, y_{N_I,I}\}$ in the area. SAE models y_I as follows:

$$y_I = \hat{y}_I + \varepsilon_I, \quad \varepsilon_I \sim N\left(0, \frac{\sigma^2}{N_I}\right), \quad (10)$$

where ε_I denotes noise with variance decreasing as the number of individuals N_I in the small area increases. The objective of SAE is estimating the denoised data \hat{y}_I .

To achieve this, we assume Eq.(11), which is a special case of Eq.(8) that can be estimated by the `resf` function:

$$y_{i,I} = \sum_{k=1}^K x_{i,I,k} \beta_k + f_{MC}(g_I) + \gamma(g_I) + \varepsilon_{i,I}, \quad \varepsilon_{i,I} \sim N(0, \sigma^2). \quad (11)$$

where g_I denotes small area ID and $x_{i,I,k}$ is an individual-level explanatory variable. Let us aggregate (take average) each term of Eq.(11). Then, the aggregated model yields Eq.(10) with

$$\hat{y}_I = \sum_{k=1}^K x_{I,k} \beta_k + f_{MC}(g_I) + \gamma(g_I), \quad (12)$$

where $x_{I,k}$ is the mean of the $x_{i,I,k}$ values in the I -th area. Eqs.(10) - (12) suggest that by estimating Eq.(11) using the `resf` function, and calculating Eq.(12) after that, we can estimate the denoised data \hat{y}_I . Similarly, the `resf_vc` function is available when estimating the denoised data \hat{y}_I while assuming the S(N)VC model with group effects instead of Eq.(11). Thus, the `resf` and `resf_vc` functions are available for SAE.

As an example, land price level in each municipality is evaluated using the land price data.

In this case, $y_{i,I}$ is the i -th land price data in the I -th municipality, and $y_I = \frac{1}{N_I} \sum_{i \in I} y_{i,I}$ is the mean price. While many municipalities include more than 10 observations, some municipalities contain only a few observations (minimum: 2); this tendency is prominent in the north area. SAE is useful to estimate the municipal land price level while reducing data uncertainty in the north area.

Here is the code for the SAE, which assumes Eq.(11) for $y_{i,I}$:

```
> meig <- meigen(coords,s_id=s_id)
> r_res <- resf(y=y, x=x,meig=meig, xgroup=s_id)
> pred <- predict0(r_res, x0=x, meig0=meig, xgroup0=s_id)
> adat <- aggregate(data.frame(y, pred$pred),by=list(s_id),mean)

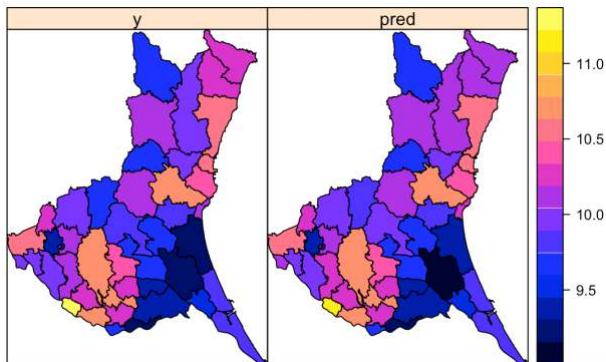
> adat[1:5,]
  Group.1      y      pred      xb sf_residual      xgroup
1  Ami-machi 10.218564 10.218639 10.349851 -0.18565423 0.05444216
2  Bando-shi  9.967086  9.960930  9.855077 -0.02577580 0.13162885
3 Chikusei-shi 9.979030  9.993724 10.316444 -0.05639032 -0.26632998
4  Daigo-machi 9.614147  9.645394  9.544478  0.13472822 -0.03381190
5  Goka-machi  9.959758 10.048423 10.289023 -0.04278297 -0.19781632
```

Therein, `s_id` (municipality name) is used in the `meigen` function to model spatially dependent group effects. The `resf` function estimates Eq.(11). Then, the `predict0` function (see Section 2.6) evaluates the individual-level predicted value $\hat{y}_{i,I} = \sum_{k=1}^K x_{i,I,k} \hat{\beta}_k + \hat{f}_{MC}(g_I) + \hat{\gamma}(g_I)$, and the `aggregate` function aggregates each term to obtain Eq.(12). `adat` summarizes the municipality level outputs. The columns are as follows:

- `y` : Sample mean: y_I
- `pred` : Denoised data: \hat{y}_I
- `xb` : Trend in the denoised data: $\sum_{k=1}^K x_{i,I,k} \hat{\beta}_k$
- `sf_residual` : Spatially dependent group effects: $\hat{f}_{MC}(g_I)$
- `xgroup` : Spatially independent group effects: $\hat{\gamma}(g_I)$

Not only, `pred`, but also `xb`, `sf_residuals`, `xgroup` are useful to understand the structure behind observations. After combining `adat` with spatial polygons of the municipalities, the estimated `y` and `pred` are be visualized as follows:

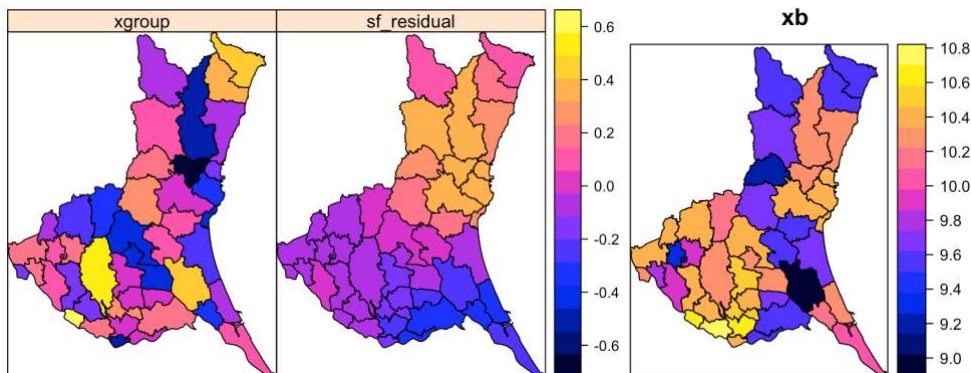
```
> splot(iba,c("y","pred"))
```



As expected, the denoised data (pred) has larger difference with the raw data (y) in the north area. The other effects can be visualized as follows:

```
> splot(iba,c("xgroup","sf_residual"))
```

```
> splot(iba,"xb", main = "xb")
```



Spatially dependent group effects (sf_residuals) are relatively smooth whereas independent group effects (xgroup) indicated large values (yellow) in two municipalities in the south-west area. They are Tsukuba and Moriya cities. These cities are popular residential areas for commuters to Tokyo, which locates about 30 km south-west from this prefecture.

Here is a code for SAE when using S(N)VC model:

```
rv_res <-resf_vc(y=y, x=x, meig=meig, xgroup=s_id, x_sel=FALSE)
```

```
pred_vc <-predict0_vc(rv_res,x0=x,meig0=meig,xgroup0=s_id)
```

```
adat_vc <-aggregate(data.frame(y, pred_vc$pred),by=list(s_id),mean)
```

```
> adat_vc[1:5,]
```

	Group.1	y	pred	xb	sf_residual	xgroup
1	Ami-machi	10.218564	10.214928	10.309744	-0.18921742	0.09440170
2	Bando-shi	9.967086	9.953377	9.712561	0.03239573	0.20841994
3	Chikusei-shi	9.979030	9.992218	10.291589	-0.07652292	-0.22284743
4	Daigo-machi	9.614147	9.647795	9.608168	0.08122060	-0.04159323
5	Goka-machi	9.959758	10.059834	10.293884	-0.01398082	-0.22006946

2.4.2. Longitudinal/panel data model

The `resf` and `resf_vc` functions are also available for longitudinal or panel data analysis with/without S(N)VC (see Yu et al., 2020). To illustrate this, we use a panel data of 48 US states from 1970 to 1986, which is published in the `plm` package (Croissant and Millo, 2008). Because our approach uses spatial coordinates by default, we added center spatial coordinates (`px` and `py`) to the panel data. Here are the first 5 rows of the data:

```
> pdat <- read.csv("Produc_s.csv")
> pdat[1:5,]
  state year region  pcpap  hwy  water  util  pc  gsp  emp unemp  px  py
1 ALABAMA 1970     6 15032.67 7325.80 1655.68 6051.20 35793.80 28418 1010.5  4.7 32.80667 -86.79113
2 ALABAMA 1971     6 15501.94 7525.94 1721.02 6254.98 37299.91 29375 1021.9  5.2 32.80667 -86.79113
3 ALABAMA 1972     6 15972.41 7765.42 1764.75 6442.23 38670.30 31303 1072.3  4.7 32.80667 -86.79113
4 ALABAMA 1973     6 16406.26 7907.66 1742.41 6756.19 40084.01 33430 1135.5  3.9 32.80667 -86.79113
5 ALABAMA 1974     6 16762.67 8025.52 1734.85 7002.29 42057.31 33749 1169.8  5.5 32.80667 -86.79113
```

Following an example in the `plm` package, logged public capital stock (`pcap`) is used as explained variables. The explained variables are logged private capital stock (`pc`), logged labor input measured by the employment in non-agricultural payrolls (`emp`), and unemployment rate (`unemp`).

```
y <- log(pdat$gsp)
x <- data.frame(log_pcpap = log(pdat$pcpap), log_pc = log(pdat$pc),
               log_emp = log(pdat$emp), unemp = pdat$unemp)
```

Because panel data has multiple observations for each individual (site/region), individual ID is needed to model individual-level spatial dependence. Here is the code extracting MEs by individuals to model spatial dependence by individuals:

```
s_id <- pdat$state
coords <- pdat[,c("px", "py")]
pmeig <- meigen(coords, s_id = s_id) # Moran eigenvectors by states
```

Currently, the following panel models can be implemented:

- Pooling model (no random effects)
- One-way (individual) random effects model
- One-way (time) random effects model
- Two-way (individual and time) random effects model

In all the models, residual spatial dependence is considered when using the `resf` function while residual spatial dependence and S(N)VC are considered when using the `resf_vc` function.

Here is the code implementing these models using the `resf` function:

```
pmod0 <-resf(y=y,x=x,meig=pmeig)           # pooling model

xgroup<-pdat$state                          # individual ID
pmod1 <-resf(y=y,x=x,meig=pmeig,xgroup=xgroup) # one-way (individual) model

xgroup<-pdat$year                          # time ID
pmod2 <-resf(y=y,x=x,meig=pmeig,xgroup=xgroup) # one-way (time) model

xgroup<-pdat[,c("state","year")]          # individual ID and time ID
pmod3 <-resf(y=y,x=x,meig=pmeig,xgroup=xgroup) # two-way model
```

Among these, BIC value of the two-way model is the smallest. The estimation result is as follows.

```
> pmod3
Call:
resf(y = y, x = x, xgroup = xgroup, meig = pmeig)

----Coefficients-----
              Estimate          SE    t_value    p_value
(Intercept)  2.266244035  0.160980415  14.0777624  0.000000e+00
log_pcap     0.006849663  0.023709501   0.2888995  7.727381e-01
log_pc       0.295108887  0.022270070  13.2513679  0.000000e+00
log_emp      0.729258595  0.024715495  29.5061293  0.000000e+00
unemp        -0.004394629  0.001066807  -4.1194237  4.220743e-05

----Variance parameter-----

Spatial effects (residuals):
              (Intercept)
random_SE      0.1568176
Moran.I/max(Moran.I)  0.3800088

Group effects:
              state      year
random_SE 0.09909044 0.02439177

----Error statistics-----
              stat
resid_SE     3.382292e-02
adjR2(cond)  9.988947e-01
rlogLik      1.406802e+03
AIC          -2.793605e+03
BIC          -2.746561e+03
```

`random_SE` values show large residual spatial dependent variation (0.157) relative to group effects (state:0.099; year:0.024) and the error variation (0.034). The estimated individual and time effects are extracted as

```
> s_g<- pmod3$b_g[[1]]
> s_g[1:5,]
      Estimate      SE    t_value
state_ALABAMA -0.05967592 0.01357188 -4.397028
state_ARIZONA  -0.05060541 0.01616867 -3.129844
state_ARKANSAS  -0.05858735 0.01403461 -4.174490
state_CALIFORNIA 0.21739354 0.01979789 10.980644
state_COLORADO  -0.13503462 0.01286928 -10.492791
> t_g<- pmod3$b_g[[2]]
> t_g[1:5,]
      Estimate      SE    t_value
year_1970 -0.006188420 0.011143668 -0.5553306
year_1971  0.002712708 0.010608721  0.2557054
year_1972  0.013174179 0.010443949  1.2614175
year_1973  0.021912546 0.010296965  2.1280588
year_1974 -0.009897092 0.009689259 -1.0214498
```

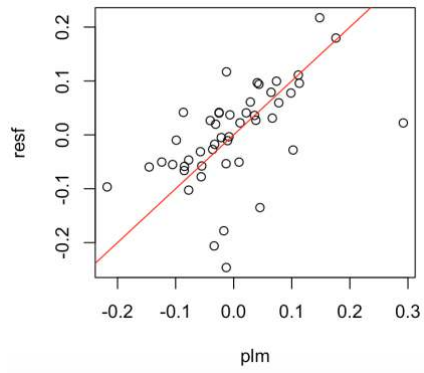
For validation, our two-way model estimates are compared with another two-way model estimated using the `plm` package. Unlike our model, it ignores residual spatial dependence. The `plm` coefficient estimates as shown below. The results are similar with our result.

```
Coefficients:
      Estimate Std. Error z-value Pr(>|z|)
(Intercept)  2.3634993  0.1389056 17.0151 < 2.2e-16 ***
log(pcap)    0.0178529  0.0233207  0.7655  0.444
log(pc)      0.2655895  0.0209824 12.6577 < 2.2e-16 ***
log(emp)     0.7448989  0.0241144 30.8902 < 2.2e-16 ***
unemp       -0.0045755  0.0010179 -4.4952  6.95e-06 ***
```

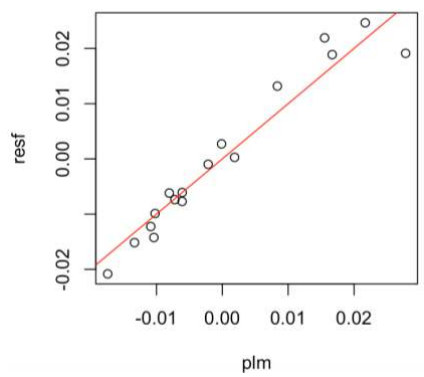
Below, estimated individual (state) and time (year) effects are compared. The result suggests that time effects estimates are quite similar whereas individual effects are less similar. This is because our model decomposes latent map pattern into spatially dependent and independent (i.e., individual) effects.

```
pm0 <- plm(log(gsp) ~ log(pcap) + log(pc) + log(emp) + unemp,
           data = pdat, effect="twoways",model="random")
s_g_plm <- ranef(pm0,"individual")
t_g_plm <- ranef(pm0,"time")
```

```
plot(s_g_plm,s_g[,1],xlab="plm",ylab="resf"); abline(0,1,col="red")# Individual effects
```



```
plot(t_g_plm,t_g[,1],xlab="plm",ylab="resf"); abline(0,1,col="red") # Time effects
```



2.5. Spatially filtered unconditional quantile regression model

While the usual (conditional) quantile regression (CQR) estimates the influence of x_k on the τ -th “conditional” quantile of y , $q_\tau(y|x_k)$, the unconditional quantile regression (UQR; Firpo et al., 2009) estimates the influence of x_k on the “unconditional” quantile of y , $q_\tau(y)$.

Suppose that y and x_k represent land price and accessibility respectively. UQR estimates the influence of accessibility on land price by quantile; it is interpretable and useful for e.g. hedonic land price analysis. By contrast, this interpretation does not hold for CQR because it estimates the influence of accessibility on conditional land prices (land price conditional on explanatory variables). Higher conditional land price does not mean higher land price, but rather, it means overprice relative to the price expected by the explanatory variables. Thus, CQR has difficulty in its interpretation in some cases including hedonic land price modeling.

Murakami and Seya (2019) developed the spatial filter UQR (SF-UQR). The SF-UQR model is formulated as follows:

$$q_\tau(y_i) = \sum_{k=1}^K x_{i,k} \beta_{k,\tau} + f_{MC,\tau}(s_i) + \varepsilon_{i,\tau}, \quad \varepsilon_{i,\tau} \sim N(0, \sigma_\tau^2), \quad (13)$$

where $q_\tau(y_i)$ is unconditional quantile for the i -th explained variable, y_i . The SF-UQR is a UQR considering spatial dependence. The `resf_qr` function estimates the SF-UQR model. The command is as follows:

```
> qr_res <- resf_qr( y = y, x = x, meig = meig, boot = TRUE )
```

If `boot = TRUE`, a semiparametric bootstrapping is performed to estimate the standard errors of the regression coefficients, and they are not calculated if `boot = FALSE`. By default, this function returns the coefficients $\beta_{k,\tau}$ estimated at the 0.1, 0.2, ..., 0.9 quantiles. An argument `tau` specifies the quantile. For example, coefficients at the 0.22 quantile are estimated by specifying `tau = 0.22`.

Here is the summary of the estimation result:

```
> qr_res
Call:
resf_qr(y = y, x = x, meig = meig, boot = TRUE)

----Coefficients-----
              tau=0.1      tau=0.2      tau=0.3      tau=0.4      tau=0.5      tau=0.6
(Intercept)  8.111826124  8.771653162  9.3782350972  9.964101377  10.1653773735  10.455828615
station      -0.100777459 -0.105972672 -0.0885991067 -0.083971480 -0.0783966978 -0.071351950
tokyo        0.004158935  0.001876297  0.0009791778 -0.001315195 -0.0008725778 -0.001530231
urban        1.515237376  1.425806626  0.9881122110  0.691648977  0.5304508318  0.385279294
flood        0.014765660  0.027267700  0.0023210183  0.012634788 -0.0023559381  0.005573913
              tau=0.7      tau=0.8      tau=0.9
(Intercept)  10.783150568  11.049070814  11.749703977
station      -0.074261737 -0.075319498 -0.099295897
tokyo        -0.002570480 -0.002997311 -0.006852662
urban        0.270011975  0.233725064  0.269818223
flood        0.001539262 -0.044487902 -0.127362823

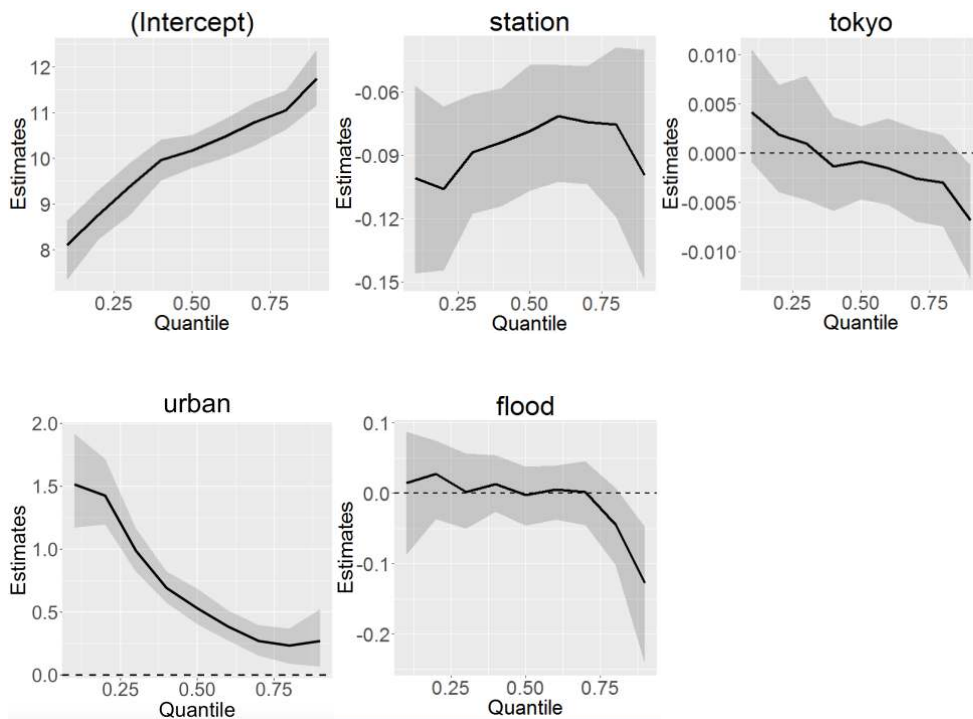
----Spatial effects (residuals)-----
              tau=0.1      tau=0.2      tau=0.3      tau=0.4      tau=0.5      tau=0.6
spcomp_SE    0.4838237  0.4714741  0.5021188  0.4907561  0.4467881  0.4929422
spcomp_Moran.I/max(Moran.I) 0.1919850  0.1598558  0.1526074  0.1569209  0.1395674  0.1203970
              tau=0.7      tau=0.8      tau=0.9
spcomp_SE    0.5261663  0.5820577  0.91940364
spcomp_Moran.I/max(Moran.I) 0.1043638  0.1138635  0.06294751

----Error statistics-----
              tau=0.1      tau=0.2      tau=0.3      tau=0.4      tau=0.5      tau=0.6      tau=0.7
resid_SE     0.9316461  0.6781962  0.5847565  0.4914106  0.4869947  0.4709563  0.5036244
quasi_adjR2(cond) 0.4383783  0.5799752  0.5738571  0.6123778  0.5561950  0.5540619  0.5188972
              tau=0.8      tau=0.9
resid_SE     0.6027896  1.0025656
quasi_adjR2(cond) 0.4649889  0.4267469
```

The estimated coefficients can be visualized using the `plot_qr` function. Here is the command to plot the estimated coefficients for the first five explanatory variables:

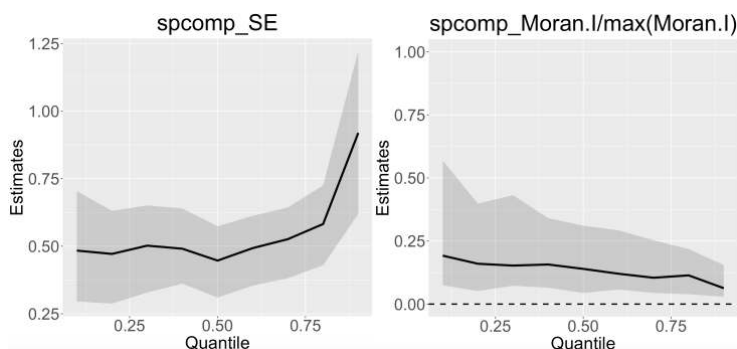
```
> plot_qr(qr_res, 1)
> plot_qr(qr_res, 2)
> plot_qr(qr_res, 3)
> plot_qr(qr_res, 4)
> plot_qr(qr_res, 5)
```

The numbers 1 to 5 specify which regression coefficients are plotted (1: intercept). The resulting plots are as below. The solid lines are estimated coefficients and the gray areas are their 95% confidence intervals.



The standard errors for the residual spatial dependent component ($spcomp_SE$) in each quantile are plotted by assigning `par = "s"` and `pnum = 1`, while the scaled ME ($Moran.I/\max(Moran.I)$), which is a measure of spatial scale by quantile ($spcomp_Moran.I/\max(Moran.I)$) are plotted by assigning `par = "s"` and `pnum = 2`. The commands and the outcomes are as follows:

```
> plot( qr_res, par = "s" , 1 )
> plot( qr_res, par = "s" , 2 )
```



Based on this result, there is larger local spatially dependent variations in higher price range.

2.6. Spatial prediction

This package provides functions for ESF/RE-ESF-based spatial interpolation minimizing the expected error variance (just like kriging) (see Murakami and Griffith, 2019c). The Nystrom extension, which is an eigen-approximation technique is used for the expected error minimization.

Note that RE-ESF approximates Gaussian process (GP) or the kriging model, which has actively been used for spatial prediction, and ESF is a special case (Murakami and Griffith, 2015). Because ESF and RE-ESF perform approximations, their spatial predictions might be less accurate relative to kriging. Instead, they are faster and available for very large samples (see Section 4).

In this tutorial, the land price data is randomly divided into two, and one is considered as observations (`dd`) and another is considered as data at unobserved sites (`md`).

```
> samp      <- sample( length( data[, 1] ), 300 )# Random sampling

> dd        <- data[ samp, ]                # Data at observed sites
> coords    <- dd[ , 1:2 ]
> y         <- dd[ ,3 ]
> x         <- dd[ ,4:7]
> xconst    <- dd[ ,1:2]

> md        <- data[-samp, ]                # Data at unobserved sites
> coords0   <- md[ , 1:2 ]
> y0        <- md[ ,3 ]
> x0        <- md[ ,4:7 ]
> xconst0   <- md[ ,1:2 ]
```

Before prediction, ME must be evaluated for both the observed and unobserved sites. `meigen` or `meigen_f` is available for the former while `meigen0` is available for the latter:

```
> meig      <- meigen( coords = coords )
> meig0     <- meigen0( meig = meig, coords0 = coords0 )
```

For ESF-based spatial interpolation, the ESF model is estimated as before. Then, data at unobserved sites are predicted using the `predict0` function. The command is as follows:

```
> mod_e     <- esf( y = y, x = x, meig = meig )           ## Model estimation
> pred_e    <- predict0( mod = mod_e, x0 = x0, meig0 = meig0 ) ## Spatial prediction
```

```

> pred_e$pred[1:6,]
      pred      xb sf_residual
1 10.66889 10.386075  0.2828139
2 10.96157 10.641180  0.3203926
4 10.25746  9.943871  0.3135856
5 10.21886 10.389411 -0.1705480
6 10.25080 10.437851 -0.1870549
7 10.39713 10.530782 -0.1336485

```

As shown above, predicted values (pred), trend (xb), and residual spatial process (sf_residual) at unobserved sites are returned. RE-ESF-based spatial interpolation is implemented in the same way:

```

> mod_re <- resf( y = y, x = x, meig = meig )           ## Model estimation
> pred_re <- predict0( mod = mod_re, x0 = x0, meig0=meig0 ) ## Spatial prediction
> pred_re$pred[1:6,]
      pred      xb sf_residual
1 10.63098 10.410279  0.2207022
2 10.94095 10.679910  0.2610403
4 10.21270  9.959378  0.2533217
5 10.24305 10.363861 -0.1208115
6 10.28607 10.418273 -0.1322042
7 10.41421 10.514778 -0.1005692

```

If `x0` is not provided, the `predict0` function interpolates only the spatial component.

```

> sf_re <- predict0( mod = mod_re, meig0=meig0 )
Note: Trend term (xb) is ignored because x0 is missing
> sf_re$pred[1:6,]
      pred sf_residual
1  NA  0.2207022
2  NA  0.2610403
3  NA  0.2533217
4  NA -0.1208115
5  NA -0.1322042
6  NA -0.1005692

```

The S(N)VC model is also available for spatial prediction using the `predict0_vc` function. When using the SNVC model, which assumes SNVC (= SVC + NVC) on `x` (`x_nvc=TRUE`) and NVC on `xconst` (`xconst_nvc=TRUE`), the command is as follows:

```

> rv_res <- resf_vc( y = y, x = x, xconst = xconst, meig = meig, x_nvc=TRUE, xconst_nvc=TRUE )
> pred_vc <- predict0_vc( mod = rv_res, x0 = x0, xconst0 = xconst0, meig0 = meig0 )

```

The predicted values are returned as

```

> pred_vc$pred[1:6,]
      pred      xb sf_residual
1 10.524921 10.524921 3.114089e-07
2 10.966584 10.966584 3.020101e-07
4 10.120867 10.120866 3.094337e-07
5  9.973893  9.973894 -3.511139e-07
6  9.985101  9.985101 -3.538379e-07
7 10.306944 10.306945 -3.514422e-07

```

This function returns the estimated S(N)VC on x as follows:

```
> pred_vc$b_vc[1:6,]
      (Intercept)      station      tokyo      urban      flood
1  10.85225 -0.1570148 -0.01077498  0.8506938 -0.01178667
2  10.85225 -0.2033687 -0.01077498  0.8690882 -0.01178667
3  10.85225 -0.1999714 -0.01077498  0.8642268 -0.01178667
4  10.85225 -0.1959116 -0.01050526  0.6350893 -0.01178667
5  10.85225 -0.2305232 -0.01050526  0.6533033 -0.01178667
6  10.85225 -0.2326697 -0.01050526  0.6941207 -0.01178667
```

It also returns the estimated NVC on xconst as follows:

```
> pred_vc$c_vc[1:6,]
      px      py
1 -5.898711e-07 1.081596e-05
2 -5.898711e-07 1.081596e-05
3 -5.898711e-07 1.081596e-05
4 -5.898711e-07 1.081596e-05
5 -5.898711e-07 1.081596e-05
6 -5.898711e-07 1.081596e-05
```

The standard errors, *t*-values, and *p*-values of the SNVCs are returned by `pred_vc$bse_vc`, `pred_vc$t_vc`, and `pred_vc$p_vc`, respectively.

When estimating SNVC or NVC at missing sites, `x0` and `xconst0` must be provided. By contrast, when estimating only SVC, the `predict0_vc` function returns the SVC estimates even if `x0` and/or `xconst0` are missing. Here is an example:

```
> rv_res2 <- resf_vc( y = y, x = x, xconst = xconst, meig = meig )
> pred_vc2 <- predict0_vc( mod = rv_res2, meig0 = meig0 )
Note: y is not predicted because x0 and/or xconst0 is missing
> pred_vc2$pred[1:6,]
NULL
> pred_vc2$b_vc[1:6,]
      (Intercept)      station      tokyo      urban      flood
1  10.40836 -0.0849332 -0.001349617  0.7276704 -0.001844808
2  10.45348 -0.0849332 -0.001349617  0.7276704 -0.001844808
3  10.44798 -0.0849332 -0.001349617  0.7276704 -0.001844808
4  10.03293 -0.0849332 -0.001349617  0.7276704 -0.001844808
5  10.03350 -0.0849332 -0.001349617  0.7276704 -0.001844808
6  10.06296 -0.0849332 -0.001349617  0.7276704 -0.001844808
> pred_vc2$c_vc[1:6,]
NULL
```

3. Low rank spatial econometric models

While Section 2 explains distance-based ESF, RE-ESF, and other spatial regression models approximating a GP (i.e., a geostatistical model), this section explains low rank spatial econometric models, approximating spatial econometric models (see Murakami et al., 2018).

Section 3.1 explains how to specify spatial connectivity matrix, and Section 3.2 and 3.3 explains the low rank spatial lag model (LSLM) and low rank spatial error model (LSEM), respectively.

3.1. Spatial weight matrix and their eigenvectors

Eigenvectors and eigenvalues of a spatial connectivity matrix, which is called spatial weight matrix or the \mathbf{W} matrix in spatial econometrics, are used for the low rank modeling. The `weigen` function is available for the eigen-decomposition.

If a shape polygon object is provided, this function returns eigenpairs of a rook adjacency-based \mathbf{W} (1 if two polygons share edge, and 0 otherwise). Here is a sample code:

```
require( spdep )
require( rgdal )
data( boston )
poly    <- readOGR( system.file( "shapes/boston_tracts.shp", package = "spData" )[ 1 ] )
weig1   <- weigen( poly )                ##### Rook adjacency-based W
```

If spatial coordinates are provided, `weigen` returns eigenpairs of the k -nearest neighbor-based \mathbf{W} by default. The comments are as follows:

```
coords  <- boston.c[ ,c( "LAT", "LON" ) ]
weig2   <- weigen( coords )              ##### 4-nearest neighbor-based W
weig3   <- weigen( coords, k = 8 )      ##### 8-nearest-neighbor-based W
```

Alternatively, the \mathbf{W} matrix can be defined based on the Delaunay triangulation. In this case, the (i, j) -th element of \mathbf{W} is 1 if the sample sites i and j share edge that is generated by the Delaunay triangulation, and 0 otherwise. This type of \mathbf{W} is used if `type = "tri"` is specified:

```
weig4   <- weigen( coords, type = "tri" ) ##### Delaunay triangulation-based W
```

An user-specified \mathbf{W} matrix is also available if the matrix is provided instead of a shape polygon object or spatial coordinates. A sample code is follows:

```
dmat <- as.matrix( rist( coords ))
cmat <- exp( -dmat )          ##### User specified  $\mathbf{W}$ 
diag(cmat)<- 0
weig5 <- weigen( cmat )
```

Note that `weigen` internally replaces all the diagonals of \mathbf{W} with zeros.

For a binary connectivity-based \mathbf{W} (`weig1` to `weig4`), $\lambda_l/\lambda_1 > 0.25$ is a standard threshold for the eigenvector selection; this criterion attempts to consider roughly 95% of the variations explaining positive spatial dependence (Griffith and Chun, 2014). This threshold is assumed by default. This threshold value can be changed. For example, $\lambda_l/\lambda_1 > 0.00$ is implemented as follows:

```
weig6 <- weigen( cmat = cmat, threshold = 0 )
```

Outputs from the `weigen` function is used to estimate low rank spatial econometric models.

3.2. Spatial regression models

3.2.1. Low rank spatial lag model

The low rank spatial lag model (LSLM) approximates the following model:

$$y_i = \beta_0 + z_i + \varepsilon_i, \quad \varepsilon_i \sim N(0, \sigma^2),$$

$$z_i = \rho \sum_{i \neq j}^K w_{i,j} z_j + \sum_{k=1}^K x_{i,k} \beta_k + u_i, \quad u_i \sim N(0, \tau^2) \quad (14)$$

where z_i is defined by the classical spatial lag model (SLM) with parameters ρ and τ^2 . Just like the original SLM, ρ takes a value between 1 and $1/\lambda_N (< 0)$. $\rho > 0$ in the presence of positive spatial dependence whereas $\rho < 0$ in the presence of negative spatial dependence. τ^2 represents the variance of the SLM-based spatial process (i.e., z_i) while σ^2 represents the variance of independent data noise.

The main differences between LSLM and SLM are as follows: (i) LSLM considers independent data noise while SLM ignores it; (ii) LSLM is faster than SLM. Due to the difference (i), the parameters estimated from LSLM and SLM is different.

The LSLM is estimated using the `lslm` function. Here is a sample code:

```

> y      <- data[, "ln_price" ]
> x      <- data[, c( "station", "tokyo", "city", "flood " ) ]
> coords <- data[, c( "px", "py" ) ]
> weig   <- weigen( coords )
> lslm_res<- lslm( y = y, x = x, weig = weig, boot = TRUE )

```

If `boot = TRUE`, a nonparametric bootstrapping is performed to estimate the 95 % confidence intervals (CIs) for the τ^2 and ρ parameters, and the direct and indirect effects, which we will explain later.

Default is FALSE. Here is the output:

```

> lslm_res
Call:
lslm(y = y, x = x, weig = weig, boot = TRUE)

----Coefficients-----
              Estimate          SE      t_value  p_value
(Intercept)  9.8435163399  0.0655499769  150.1681131  0.0000000
station     -0.0569754332  0.0060471688   -9.4218361  0.0000000
tokyo       -0.0009341669  0.0003728492   -2.5054820  0.0124876
urban        0.6054556087  0.0255012538   23.7421898  0.0000000
flood       -0.0047713534  0.0122339503   -0.3900092  0.6966657

----Spatial effects (lag)-----
              Estimates CI_lower CI_upper
sp_rho      0.5627116  0.4313565  0.6458234
sp_SE       0.4112699  0.3412107  0.4806375

----Effects estimates-----

Direct:
              Estimates  CI_lower  CI_upper  p_value
station -0.064120881 -0.07980270 -0.0510171911  0.00
tokyo   -0.001051323 -0.00193867 -0.0001665993  0.02
urban    0.681387479  0.62578040  0.7350228970  0.00
flood   -0.005369742 -0.03186009  0.0226252017  0.60

Indirect:
              Estimates  CI_lower  CI_upper  p_value
station -0.0393872899 -0.051345711 -2.619757e-02  0.00
tokyo   -0.0006457924 -0.001361761 -8.830294e-05  0.02
urban    0.4185533001  0.279064265  5.202204e-01  0.00
flood   -0.0032984511 -0.017574688  1.236365e-02  0.60

----Error statistics-----
              stat
resid_SE      0.2450407
adjR2(cond)   0.8549657
rlogLik      -240.4682106
AIC           496.9364212
BIC           532.7151916

```


`{sp_rho, sp_SE}` are estimated $\{\rho, \tau^2\}$ values. The standard error of the spatially dependent variation (`sp_SE` = 0.411) was greater than the residual standard error of 0.245. The ρ value of 0.562 suggests moderate-to-large scale spatial variation.

3.2.2. Low rank spatial error model

The low rank spatial error model (LSEM) approximates the following model:

$$y_i = \beta_0 + z_i + \varepsilon_i, \quad \varepsilon_i \sim N(0, \sigma^2),$$

$$z_i = \sum_{k=1}^K x_{i,k} \beta_k + e_i, \quad e_i = \lambda \sum_{j \neq i} w_{i,j} e_j + u_i, \quad u_i \sim N(0, \tau^2), \quad (15)$$

z_i is defined by the classical spatial error model (SEM) with parameters λ and τ^2 . λ takes a positive value in the presence of positive spatial dependence while $\rho < 0$ in the presence of negative spatial dependence. τ^2 represents the variance of the residual spatial dependence. The LSEM estimation is faster than the (maximum likelihood) estimation of the original SEM. Besides, unlike SEM, LSEM considers independent data noise, which corresponds to the nugget effect in geostatistics.

The estimated coefficients are summarized as below:

```
> lsem_res
Call:
lsem(y = y, x = x, weig = weig)

----Coefficients-----
              Estimate          SE    t_value    p_value
(Intercept) 10.013404446 0.0850568656 117.7259987 0.00000000
station     -0.079830521 0.0082862405  -9.6341062 0.00000000
tokyo       -0.001391626 0.0007416046  -1.8765064 0.06106739
urban       0.651196700 0.0310669822  20.9610543 0.00000000
flood      -0.009259069 0.0147235724  -0.6288602 0.52967818

----Spatial effects (residuals)-----
              Estimates
sp_lambda    0.9125101
sp_SE        0.2352453

----Error statistics-----
              stat
resid_SE     0.2477682
adjR2(cond)  0.8517190
rlogLik      -234.5039323
AIC          485.0078645
BIC          520.7866349
```

`{sp_lambda, sp_SE}` are estimated $\{\lambda, \tau^2\}$ values. The standard error of the spatially dependent variation (`sp_SE` = 0.235) was slightly smaller than the residual standard error of 0.248. The λ value of 0.852 suggests large scale spatial variation.

4. Tips for modeling large samples

4.1. Eigen-decomposition

Unfortunately, eigen-decomposition is known to be slow for large samples; all the methods explained in Section 2 will be slow for large samples. To overcome the limitation, the `meigen_f` function is available for fast eigen-approximation. By default, this function approximates 200 MEs. The computation is further accelerated by reducing number of eigenvectors. It is achieved by specifying `enum` by a number smaller than 200. For example, in a case with 5000 samples and `enum = 200` (default), 100, and 50, computational times are as follows:

```
> coords_test          <- cbind( rnorm( 5000 ), rnorm( 5000 ) )

-----CP time (without approximation) -----
> system.time( meig_test  <- meigen( coords = coords_test ) )
  user  system  elapsed
242.28   1.44  243.79

-----CP time (with approximation) -----
> system.time( meig_test200 <- meigen_f( coords = coords_test ) )
  user  system  elapsed
  0.37   0.00   0.38
> system.time( meig_test100 <- meigen_f( coords = coords_test, enum = 100 ) )
  user  system  elapsed
  0.15   0.00   0.16
> system.time( meig_test50  <- meigen_f( coords = coords_test, enum = 50 ) )
  user  system  elapsed
  0.08   0.00   0.08
```

Figure 3 maps the calculated 1st, 10th, and 100th eigenvectors. It is important to note that, while exact and approximated eigenvectors can have different map patterns, they always have similar spatial scales. In other words, in both cases, 1st ME describes the largest-scale map pattern, 10th ME describes the 10th largest-scale map pattern, and 100th ME describes the 100-th largest-scale patterns.

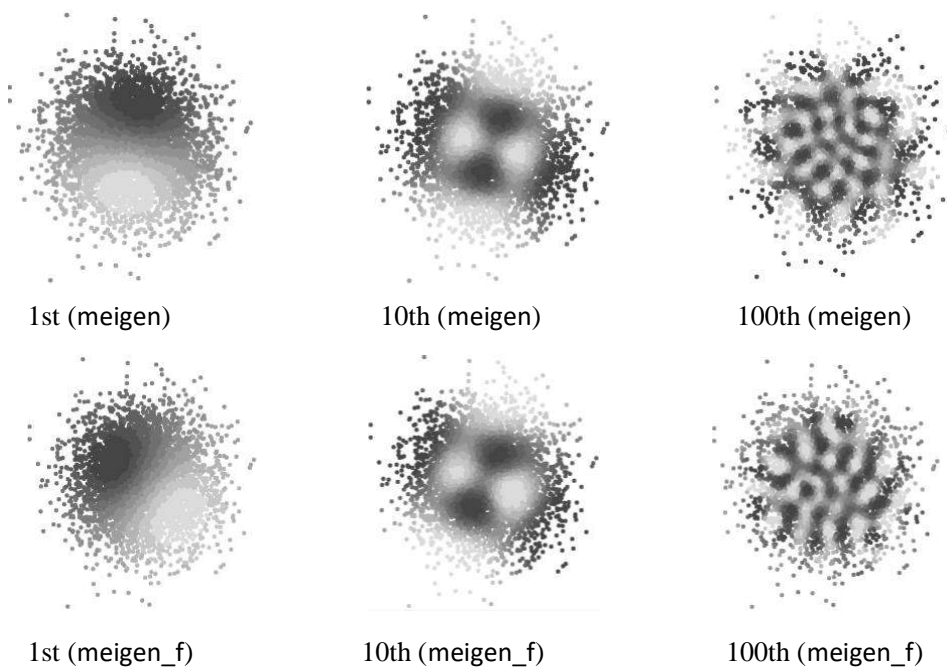


Figure 3: The 1st, 10th, and 100th MEs extracted from meigen and meigen_f

On the other hand, the weigen function implements the ARPACK routine for fast eigen-decomposition by default. The computational times with 5,000 samples and `enum = 200` (default), 100, and 50 are as follows:

```
> system.time( weig_test200 <- weigen( coords_test )
  user  system  elapsed
  9.30   0.07   9.39

> system.time( weig_test100 <- weigen( coords_test, enum = 100 ) )
  user  system  elapsed
  3.05   0.04   3.10

> system.time( weig_test50 <- weigen( coords_test, enum = 50 ) )
  user  system  elapsed
  1.19   0.03   1.22
```

4.2. Parameter estimation

The basic ESF model is estimated computationally efficiently by setting `fn = "all"` in the `esf` function. This setting is acceptable for large samples (Murakami and Griffith, 2019).

The `resf` and `resf_vc` functions estimate all the models explained in Section 2 using a fast estimation algorithm developed in Murakami and Griffith (2019b). Thus, they are available for large samples (e.g., 100,000 samples).

The SF-UQR model requires a bootstrapping to estimate confidential intervals for the coefficients. However, computational cost for the iteration does not depend on sample size. So, it is implemented computationally efficiently even for large samples.

4.3. For very large samples (e.g., millions of samples)

A computational limitation is the memory consumption of `meigen` and `meigen_f` to store the MEs. Because of the limitation, the `resf` and `resf_vc` functions are not available for very large samples (e.g., millions of samples). To overcome this limitation, the `besf` and `besf_vc` functions perform the same calculation as `resf` and `resf_vc` but without saving the whole MEs in the memory. Besides, for fast computation, these functions perform a parallel model estimation (see Murakami and Griffith, 2019c).

Here is a sample example implementing a spatial regression model using the `besf` function and a SVC model using the `besf_vc` function:

```
data <- read.csv("Data.csv")
coords <- data[,c("px", "py")]
y <- data[, "ln_price" ]
x <- data[,c( "station", "tokyo", "urban", "flood" ) ]
mod1 <- besf(y=y, x=x, coords=coords)
mod2 <- besf_vc(y=y, x=x, coords=coords)
```

I have evaluated the computational time for a SVC modeling using the `besf_vc` function using a Mac Pro (3.5 GHz, 12-Core Intel Xeon E5 processor with 64 GB memory). The `besf_vc` function took only 8.0 minutes to estimate the 7 SVCs from 1 million samples. I also confirmed that `besf_vc` took 70.3 minutes to estimate the same model from 10 million samples. `besf` and `besf_vc` are suitable for very large data analysis.

5. Future updates

Spatiotemporal models, non-Gaussian models, and extensions of the low rank spatial econometric models will be implemented in the future.

6. Reference

- Croissant, Y., and Millo, G. (2008) Panel data econometrics in R: The plm package. *Journal of statistical software*, 27(2), 1-43.
- Donegan, C., Chun, Y., Hughes, A.E. (2020) Bayesian estimation of spatial filters with Moran's eigenvectors and hierarchical shrinkage priors. *Spatial Statistics*, doi: 10.1016/j.spasta.2020.100450.
- Dray, S., Legendre, P., and Peres-Neto, P.R. (2006) Spatial modelling: a comprehensive framework for principal coordinate analysis of neighbour matrices (PCNM). *Ecological Modelling*, 196 (3), 483-493.
- Firpo, S., Fortin, N.M., and Lemieux, T. (2009) Unconditional quantile regressions. *Econometrica*, 77 (3), 953-973.
- Fotheringham, A.S., Yang, W., and Kang, W. (2017) Multiscale geographically weighted regression (MGWR). *Annals of the American Association of Geographers*, 107 (6), 1247-1265.
- Ghosh, M., and Rao, J.N.K. (1994) Small area estimation: an appraisal. *Statistical Science*. 9 (1), 55-76.
- Griffith, D.A. (2003) *Spatial autocorrelation and spatial filtering: gaining understanding through theory and scientific visualization*. Springer Science & Business Media.
- Griffith, D., & Chun, Y. (2014) Spatial autocorrelation and spatial filtering. In: *Handbook of regional science*, pp. 1477-1507. Springer.
- Legendre, P. and Legendre, L.F. (2012) *Numerical Ecology*. Elsevier.
- Murakami, D. and Griffith, D.A. (2015) Random effects specifications in eigenvector spatial filtering: a simulation study. *Journal of Geographical Systems*, 17 (4), 311-331.
- Murakami, D. and Griffith, D.A. (2019a) Eigenvector spatial filtering for large data sets: fixed and random effects approaches. *Geographical Analysis*, 51 (1), 23-49.
- Murakami, D. and Griffith, D.A. (2019b) Spatially varying coefficient modeling for large datasets: Eliminating N from spatial regressions. *Spatial Statistics*, 30, 39-64.
- Murakami, D. and Griffith, D.A. (2019c) A memory-free spatial additive mixed modeling for big spatial data. *Japan Journal of Statistics and Data Science*, doi: 10.1007/s42081-019-00063-x.
- Murakami, D., Griffith, D.A. (2020) Balancing spatial and non-spatially variations in varying coefficient modeling: a remedy for spurious correlation. *Arxiv*.

- Murakami, D. and Seya, H. (2019) Spatially filtered unconditional quantile regression. *Environmetrics*, 30 (5), e2556.
- Murakami, D., Seya, H., and Griffith, D.A. (2018) Low rank spatial econometric models. *Arxiv*, 1810.02956.
- Murakami, D., Yoshida, T., Seya, H., Griffith, D.A., and Yamagata, Y. (2017) A Moran coefficient-based mixed effects approach to investigate spatially varying relationships. *Spatial Statistics*, 19, 68-89.
- Nakaya, T., Fotheringham, S., Charlton, M., and Brunsdon, C. (2009) Semiparametric geographically weighted generalised linear modelling in GWR 4.0. *10th International Conference on GeoComputation*, Sydney.
- Reiss, P.T., and Ogden, T.R. (2009) Smoothing parameter selection for a class of semiparametric linear models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(2), 505-523.
- Snijders, T.A., and Bosker, R.J. (2011) *Multilevel analysis: An Introduction to Basic and Advanced Multilevel Modeling*. Sage.
- Tiefelsdorf, M., and Griffith, D. A. (2007) Semiparametric filtering of spatial autocorrelation: the eigenvector approach. *Environment and Planning A*, 39(5), 1193-1221.
- Wheeler, D., and Tiefelsdorf, M. (2005) Multicollinearity and correlation among local regression coefficients in geographically weighted regression. *Journal of Geographical Systems*, 7(2), 161-187.
- Yu, D., Murakami, D., Zhang, Y., Wu, X., Li, D., Wang, X., and Li, G. (2020) Investigating high-speed rail construction's support to county level regional development in China: An eigenvector based spatial filtering panel data analysis. *Transportation Research Part B: Methodological*, 133, 21-37.