# Spatial Econometrics Models with R

## Applied Spatial Econometrics

## Lecture 4 (1.5h)

Thibault LAURENT
thibault.laurent@univ-tlse1.fr

Toulouse School of Economics

15th March of 2016

Introduction

Choosing a spatial model

Estimating and interpreting a SAR model
- Estimating a SAR model
- Interpreting a SAR model

Other spatial models with R

# Preparation of the data

- Install the following packages:

```
> install.packages(c("spdep", "dismo", "RColorBrewer", "classInt",
  "GISTools", "maptools", "GeoXp", "xtable", "spgwr"))
```

- Import the data and the codes presented in the previous lectures:

```
> source("lecture3.r")
```

- Reminding about the Boston case study. The OLS model:

```
> form <- log(MEDV) ~ CRIM + ZN + INDUS + CHAS + I(NOX^2) +
  I(RM^2) +  AGE + log(DIS) + log(RAD) + TAX + PTRATIO +
  B + log(LSTAT)
> boston.lm <- lm(form, data=boston.tr@data)
```

- The OLS residuals:

```
> res.boston <- residuals(boston.lm)
```

# OLS summary

| | Dependent variable: |
|---|---|
| | log(MEDV) |
| CRIM | −0.012*** |
| | (0.001) |
| ZN | 0.0001 |
| | (0.001) |
| INDUS | 0.0002 |
| | (0.002) |
| CHAS1 | 0.091*** |
| | (0.033) |
| I(NOX^2) | −0.638*** |
| | (0.113) |
| I(RM^2) | 0.006*** |
| | (0.001) |
| AGE | 0.0001 |
| | (0.001) |
| log(DIS) | −0.191*** |
| | (0.033) |
| log(RAD) | 0.096*** |
| | (0.019) |
| TAX | −0.0004*** |
| | (0.0001) |
| PTRATIO | −0.031*** |
| | (0.005) |
| B | 0.0004*** |
| | (0.0001) |
| log(LSTAT) | −0.371*** |
| | (0.025) |
| Constant | 4.558*** |
| | (0.154) |
| Observations | 506 |
| $R^2$ | 0.806 |
| Adjusted $R^2$ | 0.801 |
| Residual Std. Error | 0.182 (df = 492) |
| F Statistic | 157.128*** (df = 13; 492) |
| *Note:* | *p<0.1; **p<0.05; ***p<0.01 |

- The linear model seems to fit well the data with a high adjusted $R^2_{adj}$.
- However, with spatial data, one may check the presence/absence of spatial autocorrelation in the residuals.
- One need to specify the spatial weight matrix.

# Specifications for $W$

Depending on the structure of your spatial data, you may use the following specifications for the spatial weights matrix:

1. $W_1$ included in `boston_listw` object (neighbouring structure: based on contiguity $\cup$ distance between $[0; 2000]m$; weights: proportionnal to the inverse distance, then row-normalized).
2. $W_2$ included in `boston_listw2` object. Neighbouring structure: based on the 5-nearest neighbours; weights: row-normalized.

```
> boston.ex_knn <- knearneigh(coord, k=5, longlat = FALSE)
> boston.ex_nb <- knn2nb(boston.ex_knn)
> boston_listw2 <- nb2listw(boston.ex_nb)
```

# Testing for spatial autocorrelation in the OLS residuals

1. OLS model is misspecified:
   - Moran test applied on the residuals:

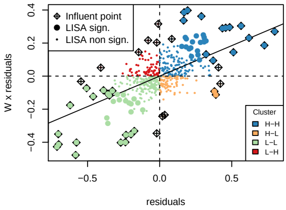
```
> morantest1 <- lm.morantest(boston.lm, boston_listw)
> morantest2 <- lm.morantest(boston.lm, boston_listw2)
```

| | $I$ | $E(I)$ | $var(I)$ | st. deviate | $p$-value |
|---|---|---|---|---|---|
| W1 | 0.36881 | -0.01593 | 0.00062 | 15.5 | $<$1e-08 |
| W2 | 0.40182 | -0.01616 | 0.00065 | 16.5 | $<$1e-08 |

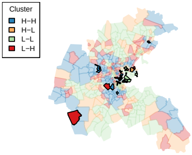
   - Spatial autocorrelation in the residuals $\Longrightarrow$ Spatial models.
2. Measuring the direct and indirect spillover effects.

# Vizualisation of local spatial autocorrelation

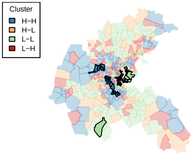
Moran plot of residuals




Influent points cluster map




Significant LISA cluster map



# Spatial models

- Mixed model with `lm`:

$$y = X\beta + WX\delta + \epsilon$$

- Spatial AutoRegressive model (SAR) with `lagsarlm`:

$$y = \rho Wy + X\beta + \epsilon$$

- Spatial Durbin Model (SDM) with `lagsarlm` (option `type="mixed"`):

$$y = \rho Wy + X\beta + WX\delta + \epsilon$$

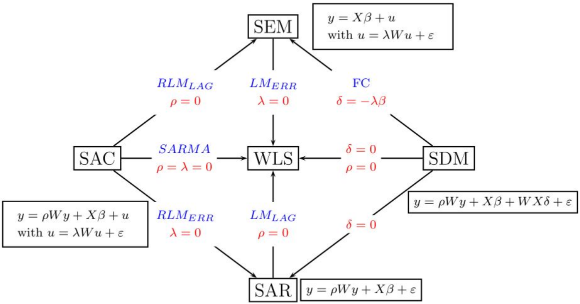
- Spatial Error Model (SEM) with `errorsarlm`:

$$y = X\beta + u, \text{ with } u = \lambda Wu + \epsilon$$

- Spatial Autoregressive Confused (SAC) with `sacsarlm`:

$$y = \rho Wy + X\beta + u, \text{ with } u = \lambda Wu + \epsilon$$

# Choice of a spatial model: Lagrange Multiplier diagnostics



# Testing strategy

1. fit an OLS model. If spatial autocorrelation in the residuals:
2. perform $LM_{ERR}$ (`test=LMerr`) and $LM_{LAG}$ (`test=LMlag`)
   - if none of them is significant, try a mixed model or keep model of step 1.
   - if only one is significant: if it is $LM_{ERR}$, keep SEM model, if it is $LM_{LAG}$, keep LAG model.
3. if both are significant, perform $RLM_{ERR}$ (`test=RLMerr`) and $RLM_{LAG}$ (`test=RLMlag`):
   - if only $RLM_{ERR}$ is significant, select SEM.
   - if only $RLM_{LAG}$ is significant, select LAG, then compare with SDM.
   - if both are significant, choose SAC, then compare with SDM.
   - if none, choose LAG (resp SEM) when $LM_{LAG}$ is more significant than $LM_{ERR}$ (resp $LM_{ERR}$ is more significant than $LM_{LAG}$).

# LM tests applied on Boston data

```
> LM.boston <- lm.LMtests(boston.lm, listw=boston_listw, test="all")
> LM.boston2 <- lm.LMtests(boston.lm, listw=boston_listw2, test="all")
```

| | df | Stat. (W1) | p-value | Stat. (W2) | p-value |
|---|---|---|---|---|---|
| LMerr | 1.00 | 205.31 | 0.00 | 232.77 | 0.00 |
| LMlag | 1.00 | 186.18 | 0.00 | 185.96 | 0.00 |
| RLMerr | 1.00 | 56.36 | 0.00 | 76.04 | 0.00 |
| RLMlag | 1.00 | 37.22 | 0.00 | 29.23 | 0.00 |
| SARMA | 2.00 | 242.53 | 0.00 | 261.99 | 0.00 |

$\Longrightarrow$ LM tests suggest to use a SAC model.

# Model selection using the AIC criterion

```
> lag.boston <- lagsarlm(form, data=boston.tr, boston_listw)
> sdm.boston <- lagsarlm(form, data=boston.tr, boston_listw,
  type="mixed")
> sac.boston <- sacsarlm(form, data=boston.tr, boston_listw)
> sem.boston <- errorsarlm(form, data=boston.tr, boston_listw)
```

| | df | AIC.W1 | AIC.W2 |
|---|---|---|---|
| SAR | 16.00 | -442.46 | -446.93 |
| SDM | 29.00 | -484.91 | -521.60 |
| SAC | 17.00 | -477.62 | -493.82 |
| SEM | 16.00 | -462.18 | -490.83 |

$\Longrightarrow$ AIC criterion suggests to use a SDM model.

# Comparison of SDM estimates with $W_1$ and $W_2$

| parameters | W1 | W2 |
|---|---|---|
| rho | 0.648403 | 0.648112 |
| | (0.042808) | (0.039717) |
| (Intercept) | 2.01864 | 1.971699 |
| | (0.281765) | (0.25638) |
| CRIM | -0.006736 | -0.005666 |
| | (0.001023) | (0.000986) |
| ZN | 0.000562 | 0.000561 |
| | (0.000534) | (5e-04) |
| INDUS | 0.000788 | -0.000713 |
| | (0.003061) | (0.002916) |
| CHAS1 | -0.041119 | -0.076568 |
| | (0.03585) | (0.033617) |
| I(NOX^2) | -0.439314 | -0.285006 |
| | (0.173732) | (0.170223) |
| I(RM^2) | 0.008822 | 0.010167 |
| | (0.001099) | (0.001068) |
| AGE | -0.001008 | -0.001389 |
| | (0.000537) | (0.000493) |
| log(DIS) | -0.204705 | -0.14379 |
| | (0.042808) | (0.039717) |
| ... | ... | ... |
| AIC | -484.91 | -521.6 |
| sigma.square | 0.0183 | 0.0171 |

Introduction

Choosing a spatial model

Estimating and interpreting a SAR model
- Estimating a SAR model
- Interpreting a SAR model

Other spatial models with R

# Estimate a SAR model by maximum likelihood

- Method considered: maximum likelihood estimation (LeSage and Pace, 2009, p.47).

$$\ln L = -(n/2)\ln(\pi\sigma^2) + \ln|I_n - \rho W| - \frac{e'e}{2\sigma^2}$$

with $e = y - y - X\beta$ and $\rho \in [min(\omega)^{-1}, max(\omega)^{-1}]$ ($\omega$ is the vector of eigenvalues of $W$)

- Goal: estimate $\rho$, $\beta$ and $\sigma$.
- Algorithm: write $\ln L$ as a 1-dimensional optimization problem (versus 3-dimensions). Then, we will deduce $\beta$ and $\sigma$.
- $\beta$ and $\sigma$ are written as functions of $\rho$. We obtain a new formula noted $Lc$ (concentrated likelihood).

# Coding the concentrated likelihood function (1)

$$Lc(\rho) = \kappa + \ln|A(\rho)| - (n/2)\ln(S(\rho))$$

```
> Lc <- function(rho) Jacobian(rho)-n/2*log(S(rho))
```

1. Compute the eigenvalues of $W$ and define the lower and upper bounds for $\rho$:

```
> W <- listw2mat(boston_listw)
> eigen.values <- eigen(W)
> bound.lambda <- 1/range(eigenw(boston_listw))
```

2. Create a functions for $A = (I_n - \rho W)$ and the so-called Jacobian ($\ln|A|$):

```
> A <- function(rho) diag(n)-rho*W
> Jacobian <- function(rho) log(prod(1-rho*eigen.values$values))
```

# Coding the concentrated likelihood function (2)

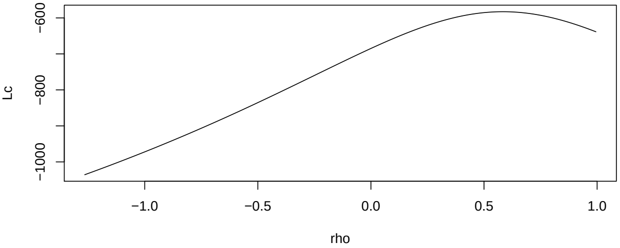
3. $S(\rho) = e_o' e_o - 2_o' e_d + \rho^2 e_d' e_d$ where $e_o$ are the residuals of an OLS of $X$ on $y$ and $e_d$ are the residuals of an OLS of $X$ on $Wy$.

```
> bo <- solve(qr(crossprod(x)), crossprod(x, y))
> eo <- y - x%*%bo
> bd <- solve(qr(crossprod(x)), crossprod(x, lag.y))
> ed <- lag.y - x%*%bd
> S <- function(rho) crossprod(eo) - 2*rho*crossprod(eo,ed) +
  rho^2*crossprod(ed)
```

# Maximizing *Lc*

4. We find the value of $\rho$ which maximises the function *Lc* by using the function `optimize`:

```
> (hat.rho <- optimize(Lc, bound.lambda, maximum = T)$maximum)
[1] 0.5061326
```



# Numerical resolution: $\hat{\beta}$ and $\hat{\sigma}$

5. Given $\hat{\rho}$, compute $\hat{\beta}$ and $\hat{\sigma}$:

```
> hat.beta <- bo - hat.rho*bd
> hat.sigma <- as.numeric(1/n*S(hat.rho))
```

**Remark**: in a big data framework, the computation of eigenvalues (step 1) could be long or infeasible. For that reason, the function `lagsarlm` provides many alternatives to approximate the Jacobian, using the R `Matrix` package for sparse matrix:

- `method="Matrix"`: the updating Cholesky decomposition,
- `method="MC"`: the Monte Carlo method MC (Barry and Pace, 1999),
- `method="Chebyshev"`: the Chebyshev method (Pace and LeSage, 2004).

# Numerical resolution: inference on $\rho$, $\beta$ and $\sigma$ (1)

6. To make inference on $\rho$, $\beta$ and $\sigma$, one needs to compute the variance/covariance matrix which is obtained by calculating the Hessian matrix (LeSage and Pace, 2009, p.54).
   - Hessian matrix can be calulated analyticaly using asymptotic formulae (when `method="eigen"`).

     ```
     > resvar <- lag.boston$resvar
     ```

   - However, this requires computing the trace of the $n \times n$ inverse matrix $A^{-1}$ which is computationally expensive. Alternatives are proposed for computing this trace which requires using sparse matrices.

     ```
     > W.sparse <-  as(as_dgRMatrix_listw(boston_listw),
       "CsparseMatrix")
     ```

# Numerical resolution: inference on $\rho$, $\beta$ and $\sigma$ (2)

- Hessian may be approximated by using a approximation of series of traces of the powered weights matrix:
  $trace(A^{-1}) = trace(I) + \rho \times trace(W) + \rho^2 \times trace(W^2) + \ldots$

```
> etr_mult <- trW(W.sparse, m = 24, type = "mult")
> system.time(lag.boston2 <- lagsarlm(form, data=boston.tr,
  boston_listw, method="Matrix", tr = etr_mult))

 user  system elapsed
 1.13    0.00    1.14
```

- One can also use Monte Carlo techniques `type="MC"` for computing the trace of $A^{-1}$.

```
> etr_MC <- trW(W.sparse, m = 24, type = "MC")
> system.time(lag.boston3 <- lagsarlm(form, data=boston.tr,
  boston_listw, method="MC", tr = etr_MC))

 user  system elapsed
 1.17    0.00    1.17
```

# Comparison of estimations methods

| parameters | eigen | Matrix.mult | MC.mc |
|---|---|---|---|
| rho | 0.5061 | 0.5061 | 0.5053 |
| | (0.0324) | (0.0324) | (0.0324) |
| (Intercept) | 2.2348 | 2.2348 | 2.2385 |
| | (0.1892) | (0.1892) | (0.1893) |
| CRIM | -0.0081 | -0.0081 | -0.0082 |
| | (0.001) | (0.001) | (0.001) |
| ZN | 5e-04 | 5e-04 | 5e-04 |
| | (4e-04) | (4e-04) | (4e-04) |
| INDUS | 5e-04 | 5e-04 | 5e-04 |
| | (0.0019) | (0.0019) | (0.0019) |
| CHAS1 | 0.0218 | 0.0218 | 0.0219 |
| | (0.0271) | (0.0271) | (0.0271) |
| I(NOX^2) | -0.3263 | -0.3263 | -0.3268 |
| | (0.0935) | (0.0935) | (0.0935) |
| I(RM^2) | 0.0064 | 0.0064 | 0.0064 |
| | (0.0011) | (0.0011) | (0.0011) |
| AGE | 2e-04 | 2e-04 | 2e-04 |
| | (4e-04) | (4e-04) | (4e-04) |
| log(DIS) | -0.1633 | -0.1633 | -0.1633 |
| | (0.0324) | (0.0324) | (0.0324) |
| ... | ... | ... | ... |
| AIC | -442.46 | -442.46 | -442.23 |
| sigma.square | 0.0218 | 0.0218 | 0.0218 |

# Numerical resolution: the tests for $\hat{\rho}$

7. Tests for $\hat{\rho}$:
   - The Likelihood Ratio test:
     ```
     > LR.test <- -2*(Lc(0)-Lc(hat.rho))
     > dchisq(LR.test, 1)
     ```
   - Asymptotic test:
     ```
     > z.value <- (hat.rho - 0)/sqrt(resvar[2,2])
     > dnorm(z.value)
     ```
   - The Wald test:
     ```
     > wald.test <- (hat.rho - 0)^2/resvar[2,2]
     > dchisq(wald.test)
     ```
8. Test spatial autocorrelation in the residuals:
   ```
   > lag.boston$LMtest
   ```

# In-sample predictions in a SAR model

9. In-sample predictions Formula in a SAR model:
   - $\hat{Y}^{TC} = (I_n - \hat{\rho}W)^{-1}X\hat{\beta}$:

```
> A.inv <- invIrW(boston_listw, hat.rho, method="solve")
> Y.TC <- A.inv%*%x%*%hat.beta
```

   - $\hat{Y}^{TS} = X\hat{\beta} + \hat{\rho}Wy$

```
> Y.TS <- x%*%hat.beta + hat.rho*W%*%y
```

   - $\hat{Y}^{BP} = \hat{Y}^{TC} - Diag(Q)^{-1}(Q - Diag(Q))(y - \hat{Y}^{TC})$, with $Q = \frac{1}{\sigma^2}(I_n - \rho W)'(I_n - \rho W)$

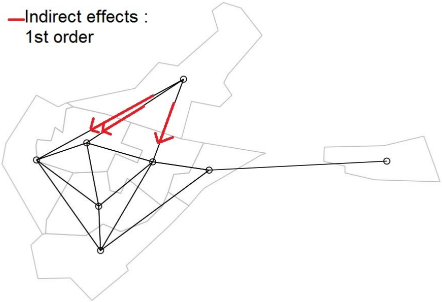
```
> Q <- crossprod(A(hat.rho))/hat.sigma
> diag.Q <- diag(n)*diag(Q)
> diag.Q.inv <- diag(n)/diag(Q)
> Y.BP <- Y.TC - diag.Q.inv%*%(Q-diag.Q)%*%(y-Y.TC)
```

# Illustration of the spatial diffusion (1)

In a SAR model: $\frac{\partial y}{\partial x_r'} = {\color{red}I_n\beta_r} + W\rho\beta_r + W^2\rho^2\beta_r + \ldots$

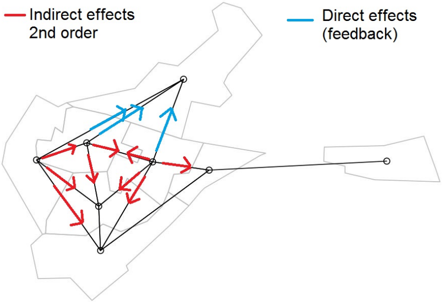
# Illustration of the spatial diffusion (2)

In a SAR model: $\frac{\partial y}{\partial x_r'} = I_n\beta_r + W\rho\beta_r + W^2\rho^2\beta_r + \ldots$



# Illustration of the spatial diffusion (3)

In a SAR model: $\frac{\partial y}{\partial x_r'} = I_n\beta_r + W\rho\beta_r + W^2\rho^2\beta_r + \ldots$



# Direct and indirect impacts

- In a SAR model, the impacts associated to a variable $r$, are deduced from the $n \times n$ matrix:

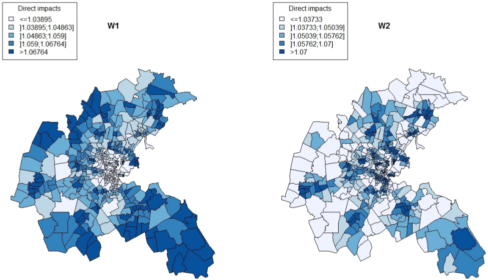
$$S_r(W) = (I - \rho W)^{-1} I_n \beta_r$$

which also requires calculating the trace of the $n \times n$ inverse matrix $A^{-1}$ (Lesage and Pace, 2009, p.38).

- For a spatial unit $i$, the **direct impact** corresponds to the $i$th diagonal element of $S_r(W)$.
- For a spatial unit $i$, the **indirect impact** corresponds to the sum of all elements of the $i$th row of $S_r(W)$, minus its diagonal.

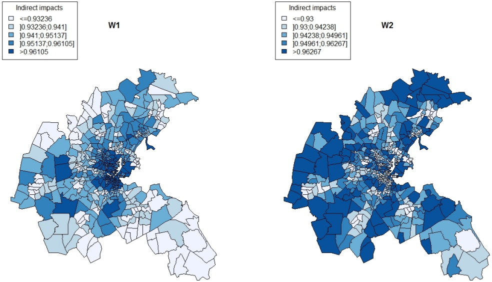
# The role of $W$ in the spatial diffusion: direct impact

To understand the role of the spatial weight matrix on the individuals direct impacts, we represent below for each $i$, with $\rho$ fixed to 0.5 $(I - \rho W)^{-1}_{ii}$, depending on $W_1$ then $W_2$.



# The role of $W$ in the spatial diffusion: indirect impact

To understand the role of the spatial weight matrix on the individuals inddirect impacts, we represent below for each $i$, with $\rho$ fixed to 0.5 $\sum_{j \neq i}(I - \rho W)^{-1}_{ij}$ (Lesage and Pace, 2009, p.36), depending on $W_1$ then $W_2$.



# Computing summary measures of impacts

Usually, one represents the average total impacts, the average indirect impacts and the average total impacts.

1. The average direct effects (diagonal elements of $S_r(W)$):
```
> direct <- sum(diag(A.inv))/n*coefficients(lag.boston)[-c(1,2)]
```
2. The average total effects (all elements of $S_r(W)$):
```
> total <- sum(A.inv)/n*coefficients(lag.boston)[-c(1,2)]
```
3. The average indirect effect (off-diagonal elements of $S_r(W)$):
```
> indirect <- total - direct
```

# Computing the impacts with R

The function `impacts` calculates the impacts:

- One can calculate it analytically (but it may take a lot of time):

```
> impacts(lag.boston, listw=boston_listw)
```

- One solution consists in approximating the trace by one of the method seen previously (the powered weights matrix or Monte-Carlo techniques):

```
> impacts.sar <- impacts(lag.boston2, tr=etr_mult)
> impacts(lag.boston3, tr=etr_MC)
```

# Computing the significance of the impacts

The function `summary.lagImpact` uses simulations to produce an empirical distribution of $\beta$, $\rho$, $\sigma$ that are needed to calculate the scalar summary measures (Lesage and Pace, 2009, p.39). Example with 200 simulations:

```
> lag.impacts <- impacts(lag.boston2, tr=etr_mult, R=200)
```

To get inferences and summary measures:

```
> res.impacts <- summary(lag.impacts, zstats=T)
```

| | Direct | Indirect | Total |
|---|---|---|---|
| CRIM | -0.00861 (***) | -0.00789 (***) | -0.0165 (***) |
| ZN | 0.00051 | 0.00047 | 0.00098 |
| INDUS | 0.00055 | 5e-04 | 0.00105 |
| CHAS1 | 0.02307 | 0.02113 | 0.0442 |
| I(NOX^2) | -0.3448 (***) | -0.31584 (***) | -0.66064 (***) |
| I(RM^2) | 0.00671 (***) | 0.00615 (***) | 0.01287 (***) |
| AGE | 0.00021 | 0.00019 | 0.00039 |
| log(DIS) | -0.17253 (***) | -0.15803 (***) | -0.33056 (***) |
| log(RAD) | 0.07893 (***) | 0.07229 (***) | 0.15122 (***) |
| TAX | -0.00032 (**) | -0.00029 (**) | -6e-04 (**) |
| PTRATIO | -0.01081 (*) | -0.00991 (*) | -0.02072 (*) |
| B | 0.00027 (**) | 0.00025 (**) | 0.00052 (**) |

# Spatial partitioning of direct, indirect and total impacts

The function `summary.lagImpact` allows partioning the impacts by order of neighbors (Lesage and Pace, 2009, p.40). Example with 6 order neighbors:

```
> lag.impacts <- impacts(lag.boston2, tr=etr_mult, R=200, Q=7)
```

To get inferences and summary measures:

```
> res.impacts <- summary(lag.impacts, zstats=T, reportQ=TRUE)
```

Application to variable `CRIM` :

| | direct | indirect | total |
|---|---|---|---|
| W0 | -0.00815 | 0.00000 | -0.00815 |
| W1 | 0.00000 | -0.00412 | -0.00412 |
| W2 | -0.00034 | -0.00175 | -0.00209 |
| W3 | -0.00006 | -0.00100 | -0.00106 |
| W4 | -0.00004 | -0.00050 | -0.00053 |
| W5 | -0.00001 | -0.00026 | -0.00027 |
| W6 | -0.00001 | -0.00013 | -0.00014 |
| Total | -0.00861 | -0.00776 | -0.01636 |

# Mixed models

$$y = X\beta + WX\delta + \epsilon$$

- Need to create manually the spatially lagged variables with function `lag.listw`.
- Perform an OLS (`lm` function) with $X$ and $WX$ as explanatory variables.
- Interpretation of the coefficients : $\hat{\beta}$ direct impact and $\hat{\delta}$ indirect impact.
- Predictions are given by function `predict`.

# SDM model

$$y = \rho W y + X\beta + WX\delta + \epsilon$$

- No need to create manually the spatially lagged variables. Use the function `lagsarlm` with option `type="mixed"`.
- If huge number of observations, use `method="Matrix"` to compute the Jacobian and `tr = etr_mult` to approximate the Hessian matrix.
- To compute the impacts, same things that for SAR model.
- For predictions, use the same than for SAR; replace $X\hat{\beta}$ by $(X\hat{\beta} + WX\hat{\delta})$.

# SEM model

$$y = X\beta + u, \text{ with } u = \lambda W u + \epsilon$$

- If huge number of observations, use `method="Matrix"` to compute the Jacobian and `tr = etr_mult` to approximate the Hessian matrix.
- No need to compute the impacts.
- For predictions, **spdep** uses in the `predict.sarlm` function, the formula : $\hat{y}^{SEM} = trend + signal$ with $trend = X\hat{\beta}$ and $signal = \hat{\lambda} W y - \hat{\lambda} W X \hat{\beta}$.

## Other methods not detailed

- Spatial model using estimation by the method of moments in package **spdep**: `gstsls` for SAR and `GMerrorsar` for SEM.
- Geographically Weighted Regression (GWR): package **spgwr** (Bivand, 2015).
- Spatial Panel Data with R: package **splm** (Millo et Piras, 2015).
- Models used in Geostatistics: kriging, filtering, smoothing. Packages **gstat** (Pebesma and Graeler, 2016) and **spatstat** (Baddeley *et al.*, 2016).