R Code for Illustration

Prof. Jude C. Hays

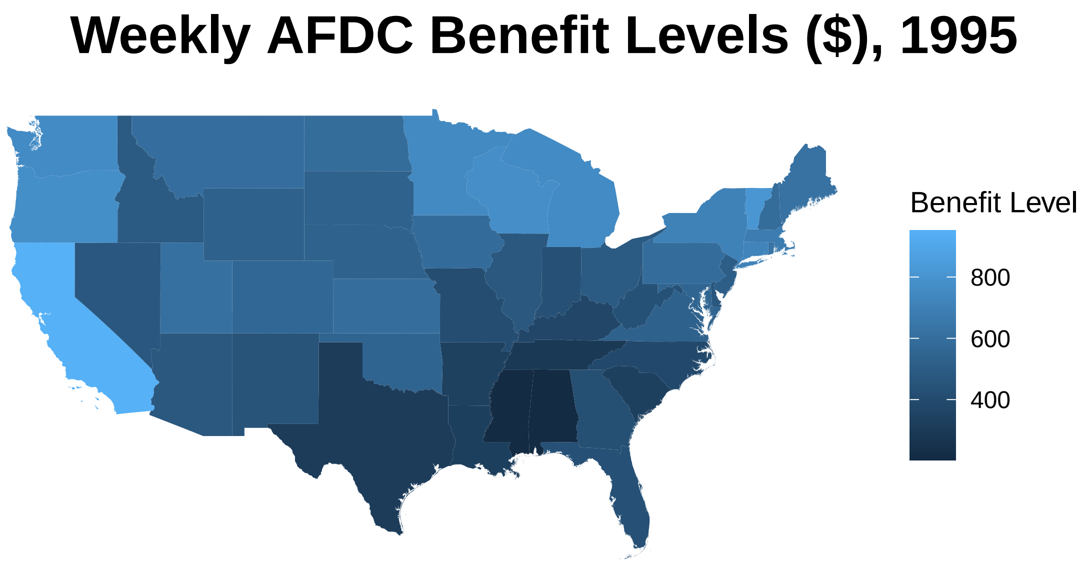
10/8/2024

Making a map with ggplot, sf, and tigris

```r
# Read the shapefile and the data
us <- read_sf("gadm36_USA_shp/gadm36_USA_1.shp")
oxford <- read.csv("oxford_data.csv")

# Check the column names for both datasets before merging
if (!("NAME_1" %in% colnames(us)) || !("name" %in% colnames(oxford))) {
    stop("Check that 'NAME_1' exists in 'us' and 'name' exists in 'oxford'")
}

# Perform the spatial join using dplyr's inner_join for
# better clarity and control
data <- inner_join(us, oxford, by = c(NAME_1 = "name"))

# Confirm the merge was successful
head(data)
```

```
## Simple feature collection with 6 features and 27 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -124.4156 ymin: 30.21725 xmax: -71.78824 ymax: 42.04973
## Geodetic CRS:  WGS 84
## # A tibble: 6 x 28
##   GID_0 NAME_0    GID_1 NAME_1 VARNAME_1 NL_NAME_1 TYPE_1 ENGTYPE_1 CC_1  HASC_1
##   <chr> <chr>     <chr> <chr>  <chr>     <chr>     <chr>  <chr>     <chr> <chr>
## 1 USA   United S~ USA.~ Alaba~ AL|Ala.   <NA>      State  State     <NA>  US.AL
## 2 USA   United S~ USA.~ Arizo~ AZ|Ariz.  <NA>      State  State     <NA>  US.AZ
## 3 USA   United S~ USA.~ Arkan~ AR|Ark.   <NA>      State  State     <NA>  US.AR
## 4 USA   United S~ USA.~ Calif~ CA|Calif. <NA>      State  State     <NA>  US.CA
## 5 USA   United S~ USA.~ Color~ CO|Colo.  <NA>      State  State     <NA>  US.CO
## 6 USA   United S~ USA.~ Conne~ CT|Conn.  <NA>      State  State     <NA>  US.CT
## # i 18 more variables: geometry <MULTIPOLYGON [°]>, statenm <chr>, ben95 <dbl>,
## #   rskpovpc <dbl>, wage95 <dbl>, instcoad <dbl>, ipcfold <dbl>,
## #   teitrend <dbl>, match <dbl>, state <int>, objectid <int>, abrev <chr>,
## #   fips <int>, lon <dbl>, lat <dbl>, shape_leng <dbl>, shape_area <dbl>,
## #   id <int>
```

```r
# Create the plot
usplot <- ggplot(data) + geom_sf(aes(fill = ben95), color = NA) +
    coord_sf(xlim = c(-125, -65), ylim = c(24, 51), expand = FALSE) +
    labs(title = "Weekly AFDC Benefit Levels ($), 1995", fill = "Benefit Level") +
    theme_minimal() + theme(plot.title = element_text(hjust = 0.5,
    size = 20, face = "bold"), plot.title.position = "plot",
    plot.margin = unit(c(1.5, 1, 1, 1), "cm"), axis.text = element_blank(),
    axis.ticks = element_blank(), panel.grid = element_blank())

# Display the plot
print(usplot)
```



Do some analysis

```r
# Load required libraries
library(spdep)
library(Matrix)
library(spatialreg)

# Load and prepare data
oxford <- read.csv("oxford_data.csv")
weights <- as.matrix(read.csv("oxford_w.csv"))
```

```r
# Function to ensure weights are properly standardized and
# transformed
standardize_weights <- function(weights_matrix) {
    weights_listw <- mat2listw(weights_matrix, row.names = NULL,
        style = "W")
    return(nb2listw(weights_listw$neighbours, style = "W"))
}

# Function to convert listw object back to matrix for
# matrix multiplication
listw_to_matrix <- function(weights_listw) {
    return(as.matrix(listw2mat(weights_listw)))  # Convert listw object back to matrix for multiplicati
}

# Function to compute the spatial lag matrix (SLX) with
# enhanced checks
compute_slx <- function(X, weights) {
    if (!is.matrix(X)) {
        stop("X must be a matrix.")
    }

    if (inherits(weights, "listw")) {
        weights <- listw_to_matrix(weights)  # Convert listw to matrix if needed
    }

    if (!is.matrix(weights)) {
        stop("Weights must be a matrix or convertible to a matrix.")
    }

    if (ncol(weights) != nrow(X)) {
        stop("The number of columns in weights must match the number of rows in X for matrix multiplica
    }

    return(weights %*% X)  # Spatial lag operation
}

# Ensure X is correctly formatted as a matrix
X <- as.matrix(oxford[, 4:9])  # Assuming the relevant variables are in columns 4:9

# Ensure weights is properly standardized and transformed
# to matrix
weights <- standardize_weights(weights)
weights_matrix <- listw_to_matrix(weights)

# Perform the spatial lag computation (SLX)
SLX <- compute_slx(X, weights_matrix)

# Function to run OLS regression and return the model (not
# the summary)
run_ols <- function(data) {
    ols_model <- lm(ben95 ~ rskpovpc + wage95 + instcoad + ipcfold +
        teitrend + match, data = data)
    return(ols_model)
}

# Run the OLS regression
ols_model <- run_ols(oxford)

# Print summary outside of the function
print(summary(ols_model))
```

```
##
## Call:
## lm(formula = ben95 ~ rskpovpc + wage95 + instcoad + ipcfold +
##     teitrend + match, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -167.97  -81.31  -11.41   54.17  377.42
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  54.5183   531.8260   0.103   0.9189
## rskpovpc     -6.5600    11.2618  -0.582   0.5634
## wage95       -0.1213     0.2269  -0.535   0.5958
## instcoad      1.5134     1.0304   1.469   0.1495
## ipcfold     621.7990   290.8709   2.138   0.0386 *
## teitrend      3.3567     1.5868   2.115   0.0405 *
## match        -4.4035     5.0007  -0.881   0.3837
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 118.7 on 41 degrees of freedom
## Multiple R-squared:  0.5297, Adjusted R-squared:  0.4609
## F-statistic: 7.698 on 6 and 41 DF,  p-value: 1.414e-05
```

```r
# LM and Robust LM tests
run_lm_tests <- function(ols_model, weights) {
    lm_tests <- lm.LMtests(ols_model, weights, test = c("LMerr",
        "LMlag", "RLMerr", "RLMlag"))
    return(summary(lm_tests))
}

# Perform LM tests on the actual OLS model object
lm_test_results <- run_lm_tests(ols_model, weights)
print(lm_test_results)
```

```
##  Lagrange multiplier diagnostics for spatial dependence
## data:
## model: lm(formula = ben95 ~ rskpovpc + wage95 + instcoad + ipcfold +
## teitrend + match, data = data)
## weights: weights
##
##        statistic parameter   p.value
## LMerr    5.84473         1 0.0156238 *
## LMlag   11.60584         1 0.0006574 ***
## RLMerr   0.71624         1 0.3973796
## RLMlag   6.47736         1 0.0109257 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Spatial Autoregressive (SAR) lag model
run_sar_model <- function(data, weights) {
    sar_model <- lagsarlm(ben95 ~ rskpovpc + wage95 + instcoad +
        ipcfold + teitrend + match, data = data, listw = weights,
        method = "eigen", zero.policy = TRUE, tol.solve = 1e-11)
    return(sar_model)
}

# Store SAR model and print its summary
sar_model <- run_sar_model(oxford, weights)
print(summary(sar_model))
```

```
##
## Call:lagsarlm(formula = ben95 ~ rskpovpc + wage95 + instcoad + ipcfold +
##     teitrend + match, data = data, listw = weights, method = "eigen",
##     zero.policy = TRUE, tol.solve = 1e-11)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -146.1977  -71.7770   -5.0836   49.8639  344.8755
##
## Type: lag
## Coefficients: (asymptotic standard errors)
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -155.41168  429.35523 -0.3620  0.71738
## rskpovpc       3.61445    8.91974  0.4052  0.68532
## wage95        -0.02553    0.18232 -0.1400  0.88864
## instcoad       1.43251    0.80614  1.7770  0.07557
## ipcfold      445.40931  226.99909  1.9622  0.04974
## teitrend       2.42686    1.26262  1.9221  0.05460
## match         -5.38937    3.90333 -1.3807  0.16737
##
## Rho: 0.53475, LR test value: 12.354, p-value: 0.00044009
## Asymptotic standard error: 0.12196
##     z-value: 4.3845, p-value: 1.1627e-05
## Wald statistic: 19.224, p-value: 1.1627e-05
##
## Log likelihood: -287.4067 for lag model
## ML residual variance (sigma squared): 8573.5, (sigma: 92.593)
## Number of observations: 48
## Number of parameters estimated: 9
## AIC: 592.81, (AIC for lm: 603.17)
## LM test for residual autocorrelation
## test value: 1.3773, p-value: 0.24056
```

```
# Moran's I test for spatial autocorrelation on residuals
moran_test_residuals <- function(residuals, weights) {
    return(moran.test(residuals, weights))
}

# Perform Moran's I tests on residuals from both OLS and
# SAR models
ols_resid_moran <- moran_test_residuals(resid(ols_model), weights)
sar_resid_moran <- moran_test_residuals(resid(sar_model), weights)

print(ols_resid_moran)
```

```
##
##  Moran I test under randomisation
##
## data:  residuals
## weights: weights
##
## Moran I statistic standard deviate = 2.8002, p-value = 0.002554
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##       0.246473800      -0.021276596       0.009143082
```

```
print(sar_resid_moran)
```

```
##
##  Moran I test under randomisation
##
## data:  residuals
## weights: weights
##
## Moran I statistic standard deviate = -0.47255, p-value = 0.6817
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##      -0.065960243      -0.021276596       0.008941381
```

```
# Corrected Likelihood Ratio Test for SAR vs OLS
compute_likelihood_ratio <- function(sar_model, ols_model) {
    logLik_ols <- logLik(ols_model)
    logLik_sar <- logLik(sar_model)
    lr_statistic <- 2 * (logLik_sar - logLik_ols)

    # Degrees of freedom: Difference in the number of
    # estimated parameters between SAR and OLS
    df <- attr(logLik_sar, "df") - attr(logLik_ols, "df")

    # p-value for the chi-square test
    p_value <- pchisq(lr_statistic, df = df, lower.tail = FALSE)

    return(list(lr_statistic = lr_statistic, p_value = p_value))
}
```

```r
# Run the Likelihood Ratio Test using the actual SAR model
# object (not summary)
lr_test_results <- compute_likelihood_ratio(sar_model, ols_model)

# Print the likelihood ratio test statistic and p-value
print(paste("LR Statistic:", lr_test_results$lr_statistic))
```

```
## [1] "LR Statistic: 12.3537872755127"
```

```r
print(paste("P-value:", lr_test_results$p_value))
```

```
## [1] "P-value: 0.000440092774796709"
```

```r
# Wald test for SAR model (manually calculated)
run_wald_test <- function(model, param_name) {
    # Get the coefficient and variance-covariance matrix of
    # the model
    coef_val <- coef(model)[param_name]
    vcov_matrix <- vcov(model)

    # Extract the variance for the parameter of interest
    param_var <- vcov_matrix[param_name, param_name]

    # Compute the Wald statistic
    wald_stat <- (coef_val^2)/param_var

    # Compute p-value for the chi-square test (df = 1)
    p_value <- pchisq(wald_stat, df = 1, lower.tail = FALSE)

    return(list(wald_stat = wald_stat, p_value = p_value))
}

# Perform the Wald test for the spatial lag parameter
# ('rho')
wald_test_results <- run_wald_test(sar_model, "rho")
print(paste("Wald Statistic:", wald_test_results$wald_stat))
```

```
## [1] "Wald Statistic: 19.2235833013209"
```

```r
print(paste("P-value:", wald_test_results$p_value))
```

```
## [1] "P-value: 1.1626813098791e-05"
```

```r
# Spatial Error Model (SEM)
run_sem_model <- function(data, weights) {
    sem_model <- errorsarlm(ben95 ~ rskpovpc + wage95 + instcoad +
        ipcfold + teitrend + match, data = data, listw = weights,
        method = "eigen", tol.solve = 1e-20)
    return(summary(sem_model))
}
```

```r
sem_summary <- run_sem_model(oxford, weights)
print(sem_summary)
```

```
##
## Call:errorsarlm(formula = ben95 ~ rskpovpc + wage95 + instcoad + ipcfold +
##     teitrend + match, data = data, listw = weights, method = "eigen",
##     tol.solve = 1e-20)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -143.691  -56.049  -12.929   38.018  364.777
##
## Type: error
## Coefficients: (asymptotic standard errors)
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) 681.90010  472.06382  1.4445  0.14860
## rskpovpc      3.35826   10.05688  0.3339  0.73843
## wage95       -0.34561    0.24292 -1.4227  0.15481
## instcoad      1.69858    0.82069  2.0697  0.03848
## ipcfold     260.97438  238.08142  1.0962  0.27301
## teitrend      2.92563    1.21123  2.4154  0.01572
## match        -6.90918    4.09402 -1.6876  0.09148
##
## Lambda: 0.57089, LR test value: 8.4244, p-value: 0.0037022
## Asymptotic standard error: 0.13366
##     z-value: 4.2713, p-value: 1.943e-05
## Wald statistic: 18.244, p-value: 1.943e-05
##
## Log likelihood: -289.3714 for error model
## ML residual variance (sigma squared): 9181.2, (sigma: 95.819)
## Number of observations: 48
## Number of parameters estimated: 9
## AIC: NA (not available for weighted model), (AIC for lm: 603.17)
```

```r
# Spatial Lag X Model (SLX)
run_slx_model <- function(data, SLX) {
    slx_model <- lm(ben95 ~ rskpovpc + wage95 + instcoad + ipcfold +
        teitrend + match + SLX[, 1], data = data)
    return(summary(slx_model))
}

slx_summary <- run_slx_model(oxford, SLX)
print(slx_summary)
```

```
##
## Call:
## lm(formula = ben95 ~ rskpovpc + wage95 + instcoad + ipcfold +
##     teitrend + match + SLX[, 1], data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -175.99  -83.02   -3.43   64.77  345.73
```

```
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -108.9055   548.9518  -0.198   0.8437
## rskpovpc      -4.7299    11.3352  -0.417   0.6787
## wage95        -0.3769     0.3184  -1.184   0.2435
## instcoad       1.7645     1.0501   1.680   0.1007
## ipcfold      642.4551   290.3816   2.212   0.0327 *
## teitrend       3.4489     1.5831   2.179   0.0353 *
## match         -6.3192     5.2584  -1.202   0.2365
## SLX[, 1]       0.4531     0.3975   1.140   0.2612
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 118.2 on 40 degrees of freedom
## Multiple R-squared:  0.5445, Adjusted R-squared:  0.4648
## F-statistic: 6.832 on 7 and 40 DF,  p-value: 2.367e-05
```

```
# Spatial Durbin Model (SDM)
run_sdm_model <- function(data, SLX, weights) {
    sdm_model <- lagsarlm(ben95 ~ rskpovpc + wage95 + instcoad +
        ipcfold + teitrend + match + SLX[, 1], data = data, listw = weights,
        method = "eigen", zero.policy = TRUE, tol.solve = 1e-11)
    return(summary(sdm_model))
}

sdm_summary <- run_sdm_model(oxford, SLX, weights)
print(sdm_summary)
```

```
##
## Call:lagsarlm(formula = ben95 ~ rskpovpc + wage95 + instcoad + ipcfold +
##     teitrend + match + SLX[, 1], data = data, listw = weights,
##     method = "eigen", zero.policy = TRUE, tol.solve = 1e-11)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -152.8641  -83.7010   -7.7707   57.1996  320.1434
##
## Type: lag
## Coefficients: (asymptotic standard errors)
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept) -282.36288  431.85099 -0.6538  0.51321
## rskpovpc       4.92881    8.90467  0.5535  0.57992
## wage95        -0.22988    0.25226 -0.9113  0.36215
## instcoad       1.63307    0.81564  2.0022  0.04526
## ipcfold      464.23724  224.54421  2.0675  0.03869
## teitrend       2.51288    1.25524  2.0019  0.04529
## match         -6.89774    4.07280 -1.6936  0.09034
## SLX[, 1]       0.35993    0.31067  1.1586  0.24664
##
## Rho: 0.52742, LR test value: 12.168, p-value: 0.00048604
## Asymptotic standard error: 0.12073
##     z-value: 4.3687, p-value: 1.2498e-05
## Wald statistic: 19.086, p-value: 1.2498e-05
```

```
##
## Log likelihood: -286.7323 for lag model
## ML residual variance (sigma squared): 8357.1, (sigma: 91.417)
## Number of observations: 48
## Number of parameters estimated: 10
## AIC: 593.46, (AIC for lm: 603.63)
## LM test for residual autocorrelation
## test value: 0.62235, p-value: 0.43018
```

```
# Spatial Durbin Error Model (SDEM)
run_sdem_model <- function(data, SLX, weights) {
    sdem_model <- errorsarlm(ben95 ~ rskpovpc + wage95 + instcoad +
        ipcfold + teitrend + match + SLX[, 1], data = data, listw = weights,
        method = "eigen", tol.solve = 1e-20)
    return(summary(sdem_model))
}

sdem_summary <- run_sdem_model(oxford, SLX, weights)
print(sdem_summary)
```

```
##
## Call:errorsarlm(formula = ben95 ~ rskpovpc + wage95 + instcoad + ipcfold +
##     teitrend + match + SLX[, 1], data = data, listw = weights,
##     method = "eigen", tol.solve = 1e-20)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -150.209  -55.713  -23.057   61.516  330.302
##
## Type: error
## Coefficients: (asymptotic standard errors)
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) 333.69903  542.08644  0.6156 0.538170
## rskpovpc      3.40981    9.91222  0.3440 0.730846
## wage95       -0.43754    0.25078 -1.7447 0.081031
## instcoad      1.90973    0.82885  2.3041 0.021220
## ipcfold     266.27155  234.96038  1.1333 0.257104
## teitrend      3.16421    1.20930  2.6166 0.008882
## match        -8.20089    4.18375 -1.9602 0.049975
## SLX[, 1]      0.44797    0.36768  1.2184 0.223088
##
## Lambda: 0.56241, LR test value: 8.3491, p-value: 0.0038588
## Asymptotic standard error: 0.13531
##     z-value: 4.1564, p-value: 3.2333e-05
## Wald statistic: 17.276, p-value: 3.2333e-05
##
## Log likelihood: -288.642 for error model
## ML residual variance (sigma squared): 8935.5, (sigma: 94.528)
## Number of observations: 48
## Number of parameters estimated: 10
## AIC: NA (not available for weighted model), (AIC for lm: 603.63)
```

```r
# Spatial Autocorrelation Combined Model (SAC)
run_sac_model <- function(data, weights) {
    sac_model <- sacsarlm(ben95 ~ rskpovpc + wage95 + instcoad +
        ipcfold + teitrend + match, data = data, listw = weights,
        listw2 = NULL, type = "sac", method = "eigen", tol.solve = 1e-20)
    return(summary(sac_model))
}

sac_summary <- run_sac_model(oxford, weights)
print(sac_summary)
```

```
##
## Call:sacsarlm(formula = ben95 ~ rskpovpc + wage95 + instcoad + ipcfold +
##     teitrend + match, data = data, listw = weights, listw2 = NULL,
##     type = "sac", method = "eigen", tol.solve = 1e-20)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -177.1505  -59.2969   -2.5813   59.1033  290.5794
##
## Type: sac
## Coefficients: (asymptotic standard errors)
##                Estimate  Std. Error z value Pr(>|z|)
## (Intercept) -311.892244  359.478505 -0.8676 0.385600
## rskpovpc       4.290139    7.578558  0.5661 0.571333
## wage95         0.049594    0.135540  0.3659 0.714441
## instcoad       1.254842    0.737216  1.7021 0.088730
## ipcfold      523.968000  202.524490  2.5872 0.009676
## teitrend       1.504001    1.262202  1.1916 0.233430
## match         -5.265531    3.526568 -1.4931 0.135410
##
## Rho: 0.69213
## Asymptotic standard error: 0.12361
##     z-value: 5.5994, p-value: 2.1505e-08
## Lambda: -0.50962
## Asymptotic standard error: 0.28605
##     z-value: -1.7816, p-value: 0.074816
##
## LR test value: 14.302, p-value: 0.0007841
##
## Log likelihood: -286.4326 for sac model
## ML residual variance (sigma squared): 7238, (sigma: 85.076)
## Number of observations: 48
## Number of parameters estimated: 10
## AIC: 592.87, (AIC for lm: 603.17)
```

```r
# Generalized Nested Spatial Model (GNS)
run_gns_model <- function(data, SLX, weights) {
    gns_model <- sacsarlm(ben95 ~ rskpovpc + wage95 + instcoad +
        ipcfold + teitrend + match + SLX[, 1], data = data, listw = weights,
        listw2 = NULL, type = "sac", method = "eigen", tol.solve = 1e-20)
    return(summary(gns_model))
}
```

```r
gns_summary <- run_gns_model(oxford, SLX, weights)
print(gns_summary)
```

```
##
## Call:sacsarlm(formula = ben95 ~ rskpovpc + wage95 + instcoad + ipcfold +
##     teitrend + match + SLX[, 1], data = data, listw = weights,
##     listw2 = NULL, type = "sac", method = "eigen", tol.solve = 1e-20)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -177.2464  -62.4135   -1.3422   60.9409  289.0547
##
## Type: sac
## Coefficients: (asymptotic standard errors)
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -352.73716  372.23620 -0.9476  0.34332
## rskpovpc       5.11772    7.88368  0.6492  0.51624
## wage95        -0.11201    0.26147 -0.4284  0.66836
## instcoad       1.37603    0.76571  1.7971  0.07232
## ipcfold      530.83959  206.89528  2.5657  0.01030
## teitrend       1.65954    1.27313  1.3035  0.19240
## match         -6.10553    3.77722 -1.6164  0.10601
## SLX[, 1]       0.22251    0.30751  0.7236  0.46932
##
## Rho: 0.66812
## Asymptotic standard error: 0.13572
##     z-value: 4.9226, p-value: 8.5394e-07
## Lambda: -0.42739
## Asymptotic standard error: 0.30165
##     z-value: -1.4168, p-value: 0.15653
##
## LR test value: 13.245, p-value: 0.0013298
##
## Log likelihood: -286.1938 for sac model
## ML residual variance (sigma squared): 7387.2, (sigma: 85.949)
## Number of observations: 48
## Number of parameters estimated: 11
## AIC: 594.39, (AIC for lm: 603.63)
```