Spatial Regression Analysis in R

based on Luc Anselin's Workbook

Jens Bruno Wittek

03.02.2015

Table of Contents

GeoDa

- free software for spatial data analysis
- graphical and interactive interface
- functionality
  - spatial data manipulation
  - spatial data transformation
  - mapping
  - exploratory data analysis
  - spatial autocorrelation
  - spatial regression

Spatial File Formats

Shape Files

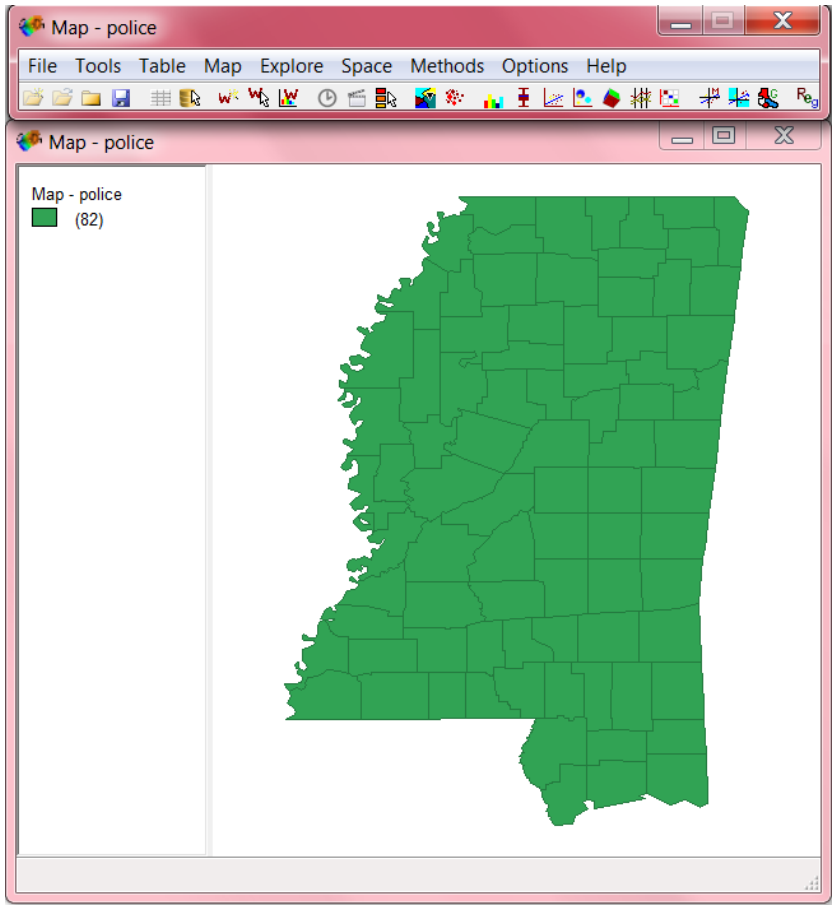
- popular across GIS software
- vector data - points, lines and polygons
- file extensions: .shp, .shx, .dbf

Spatial Weights Files

- text files
- suitable for R
- information about neighbouring regions
- file extensions: .gal (neighbours), .gwt (distances)

Read Shape Files (.shp) with GeoDa

File - New Project From - ESRI Shapefile - police.shp[1]



[1] Source of Data: https://geodacenter.asu.edu/sdata

Police Data

Police Data: 82 Counties of Mississippi

Table - police

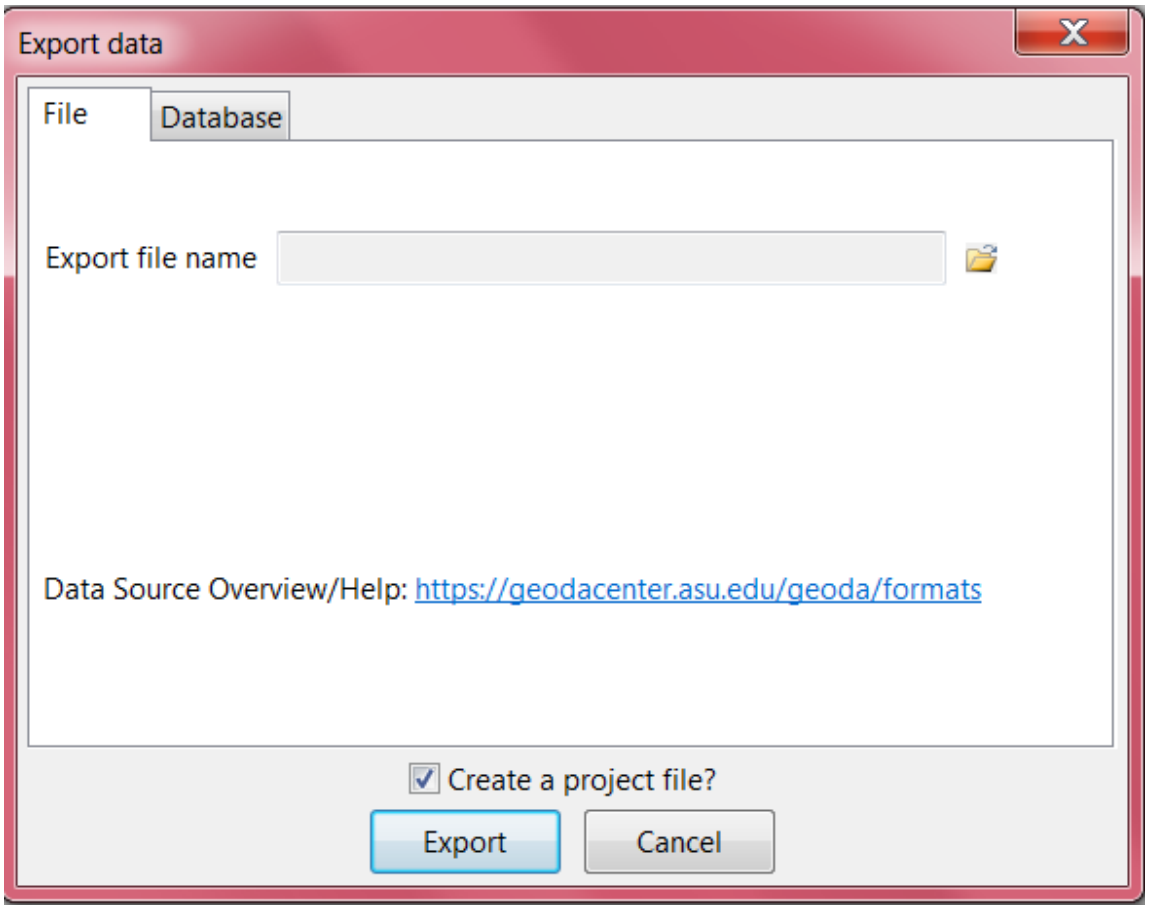
File Tools Table Map Explore Space Methods Options Help

Table - police

	AREA	PERIMETER	CNTY_	CNTY_ID	NAME	STATE_NAME	STA
1	0.105	1.401	2129	2129	Alcorn	Mississippi	28
2	0.111	1.485	2130	2130	Tishomingo	Mississippi	28
3	0.116	1.519	2131	2131	Tippah	Mississippi	28
4	0.105	1.478	2132	2132	Benton	Mississippi	28
5	0.127	1.774	2133	2133	De Soto	Mississippi	28
6	0.181	1.911	2134	2134	Marshall	Mississippi	28
7	0.119	2.146	2155	2155	Tunica	Mississippi	28
8	0.103	1.571	2171	2171	Tate	Mississippi	28
9	0.106	1.394	2176	2176	Prentiss	Mississippi	28
10	0.109	1.493	2199	2199	Union	Mississippi	28
11	0.175	1.713	2209	2209	Lafayette	Mississippi	28
12	0.184	1.745	2210	2210	Panola	Mississippi	28
13	0.151	2.451	2214	2214	Coahoma	Mississippi	28
14	0.116	1.447	2218	2218	Lee	Mississippi	28
15	0.103	1.480	2220	2220	Quitman	Mississippi	28
16	0.136	1.473	2233	2233	Itawamba	Mississippi	28
17	0.127	1.450	2239	2239	Pontotoc	Mississippi	28

Export Data (.csv) from GeoDa

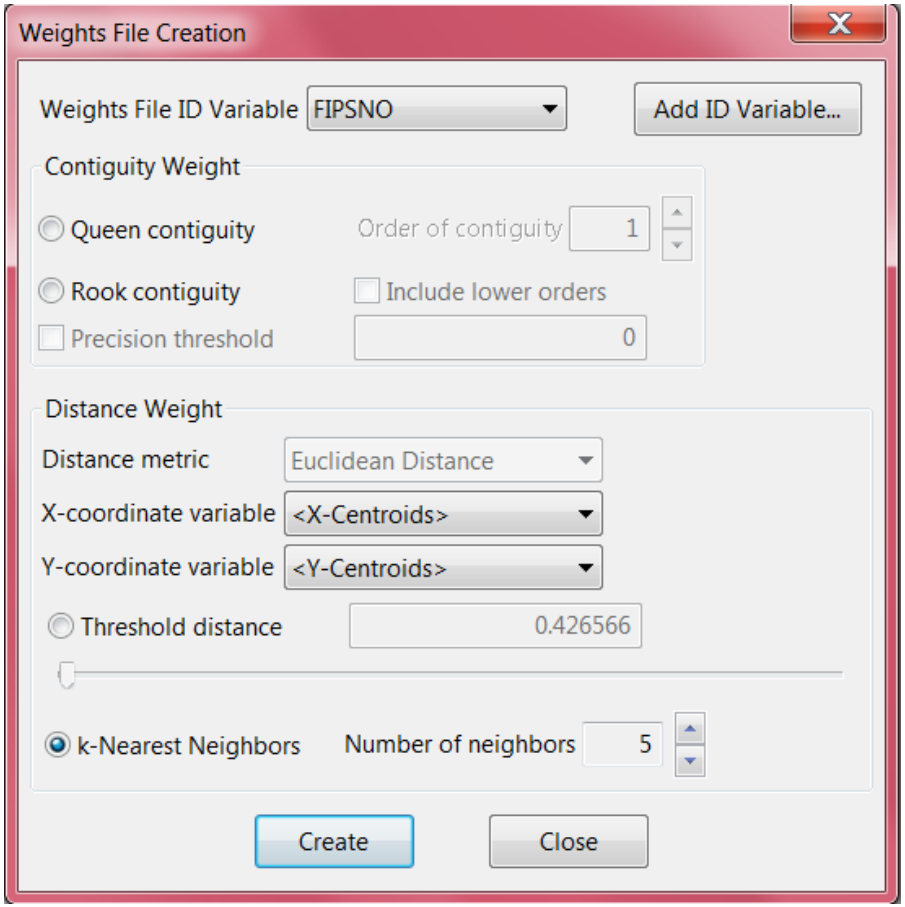
File - *Export* - set options - *csv*[2] - *Save* - *Export*



[2] The workbook suggests ASCII .txt format which is no longer supported

Create Spatial Weights File (.gal / .gwt) with GeoDa

Tools - *Weights* - *Create* - set options - *Create* - *Save*



Import Data (.csv) into R

load package for spatial econometrics (install it before)

- **spdep**: spatial dependence

```
library(spdep)
```

import data into R

```
police=read.table("./police/police.csv",sep=",",header=TRUE)
police=police[5:dim(police)[2]]
summary(police)
attach(police)
```

Import Spatial Weights File (.gal) into R

read spatial weights file into R: neighbour list object

```
polgal = read.gal("./police/policerook.gal",region.id=NULL,
override.id=TRUE)
summary(polgal)

## Neighbour list object:
## Number of regions: 82
## Number of nonzero links: 402
## Percentage nonzero weights: 5.978584
## Average number of links: 4.902439
## Link number distribution:
##  2  3  4  5  6  7
##  1 13 16 23 21  8
## 1 least connected region:
## 28045 with 2 links
## 8 most connected regions:
## 28145 28071 28135 28043 28155 28007 28163 28085 with 7 link
```

Import Spatial Weights File (.gwt) into R

read spatial weights file into R: neighbour list object

- the `region.id` has to be in the environment

```
polgwt = read.gwt2nb("./police/policek5.gwt",region.id=FIPSNO)
print(polgwt)

## Neighbour list object:
## Number of regions: 82
## Number of nonzero links: 410
## Percentage nonzero weights: 6.097561
## Average number of links: 5
## Non-symmetric neighbours list

print(is.symmetric.nb(polgal))

## [1] TRUE

detach(police)
```

Spatial Weights

- rook (cf. chess) contiguity: shared borders
- queen contiguity: shared borders or vertices
- grid versus torus layout
- **`cell2nb`** creates neighbour list objects (sparse)
- **`nb2listw`** converts it to a spatial weights objects
- spatial weights objects contain more information

Create Neighbour List Objects - 4 × 4 Rook Grid

```
rook4x4 = cell2nb(4,4,type="rook",torus=FALSE)
summary(rook4x4)

## Neighbour list object:
## Number of regions: 16
## Number of nonzero links: 48
## Percentage nonzero weights: 18.75
## Average number of links: 3
## Link number distribution:
## 2 3 4
## 4 8 4
## 4 least connected regions:
## 1:1 4:1 1:4 4:4 with 2 links
## 4 most connected regions:
## 2:2 3:2 2:3 3:3 with 4 links
```

Create Neighbour List Objects - 4x4 Rook Torus

```
rook4x4t = cell2nb(4,4,type="rook",torus=TRUE)
rook4x4t

## Neighbour list object:
## Number of regions: 16
## Number of nonzero links: 64
## Percentage nonzero weights: 25
## Average number of links: 4
```

Create Neighbour List Objects - 5x10 Queen Grid

```
queen5x10 = cell2nb(5,10,type="queen",torus=FALSE)
summary(queen5x10)

## Neighbour list object:
## Number of regions: 50
## Number of nonzero links: 314
## Percentage nonzero weights: 12.56
## Average number of links: 6.28
## Link number distribution:
##  3  5  8
##  4 22 24
## 4 least connected regions:
## 1:1 5:1 1:10 5:10 with 3 links
## 24 most connected regions:
## 2:2 3:2 4:2 2:3 3:3 4:3 2:4 3:4 4:4 2:5 3:5 4:5 2:6 3:6 4:6
```

Columbus Data

Columbus Data: 49 neighbourhoods in Columbus, OH

```
data(columbus)
attach(columbus)
str(columbus,list.len=10)
```

```
## 'data.frame': 49 obs. of  22 variables:
##  $ AREA      : num  0.3094 0.2593 0.1925 0.0838 0.4889 ...
##  $ PERIMETER : num  2.44 2.24 2.19 1.43 3 ...
##  $ COLUMBUS. : int  2 3 4 5 6 7 8 9 10 11 ...
##  $ COLUMBUS.I: int  5 1 6 2 7 8 4 3 18 10 ...
##  $ POLYID    : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ NEIG      : int  5 1 6 2 7 8 4 3 18 10 ...
##  $ HOVAL     : num  80.5 44.6 26.4 33.2 23.2 ...
##  $ INC       : num  19.53 21.23 15.96 4.48 11.25 ...
##  $ CRIME     : num  15.7 18.8 30.6 32.4 50.7 ...
##  $ OPEN      : num  2.851 5.297 4.535 0.394 0.406 ...
##   [list output truncated]
```

Create Spatial Weights Objects

convert neighbour list object to spatial weights object

```
colqueen = nb2listw(col.gal.nb)
print(colqueen)
```

```
## Characteristics of weights list object:
## Neighbour list object:
## Number of regions: 49
## Number of nonzero links: 230
## Percentage nonzero weights: 9.579342
## Average number of links: 4.693878
## Weights style: W
## Weights constants summary:
##    n   nn S0       S1       S2
## W 49 2401 49 23.48489 204.6687
```

Spatial Weights

```
head(colqueen$weights,5)
```

```
## [[1]]
## [1] 0.5 0.5
## [[2]]
## [1] 0.3333333 0.3333333 0.3333333
## [[3]]
## [1] 0.25 0.25 0.25 0.25
## [[4]]
## [1] 0.25 0.25 0.25 0.25
## [[5]]
## [1] 0.1428571 0.1428571 0.1428571 0.1428571 0.1428571
## [6] 0.1428571 0.1428571
```

Moran's Test - Hypotheses and Moran's I

- Moran's test for spatial dependence
- $H_0 : I = 0$ (no spatial dependence)
- $H_1 : I \neq 0$ (spatial dependence)
- Test statistic - Moran's I: $\hat{I}$

$$\hat{I} = \frac{n}{\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(y_i - \bar{y})(y_j - \bar{y})}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

$$= \frac{n}{\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}} \frac{z'Wz}{z'z} \in [-1; 1]$$

- left fraction equals 1 if spatial weights of neighbours of each region sum up to 1

Moran's I - Properties

- under normal approximation expected value and variance only depend on spatial weights

properties of Moran's I

$$E(\hat{I}) = \frac{-1}{n-1} \qquad E(\hat{I}^2) = \frac{n^2 S_1 - n S_2 + 3 S_0^2}{(n-1)(n+1) S_0^2}$$

$$S_0 = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} \qquad S_1 = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (w_{ij} + w_{ji})^2$$

$$S_2 = \sum_{i=1}^{n} (w_{i*} + w_{j*})^2 \qquad \mathrm{Var}(\hat{I}) = E(\hat{I}^2) - (E(\hat{I}))^2$$

Moran's Test - Test Statistic and Decision Rule

test statistic: standardized Moran's I

$$\hat{I}_z = \frac{\hat{I} - E(\hat{I})}{\sqrt{\text{Var}(\hat{I})}}$$

- decision rule (two-sided): discard H_0 if $|\hat{I}_z| > N(0,1)_{1-\alpha/2} \approx 1.96$ $(\alpha = 0.05)$

Moran's Test - Normal Approximation

Moran's test of CRIME-variable

- `randomisation=FALSE` uses normal approximation

```
moran.test(CRIME,listw=colqueen,randomisation=FALSE,
alternative="two.sided")
```

```
##  Moran's I test under normality
## data:  CRIME
## weights: colqueen
## Moran I statistic standard deviate = 5.3818, p-value
## = 7.374e-08
## alternative hypothesis: two.sided
## sample estimates:
## Moran I statistic       Expectation          Variance
##      0.485770914     -0.020833333      0.008860962
```

decision: discard H_0 - spatial dependence

Moran's Test - Permutation

- options to calculate variance: normal approximation, randomisation (default), permutation (different command)

variance calculated by permutation:

```
set.seed(123456)
(morpermCRIME=moran.mc(CRIME,listw=colqueen,nsim=999))

##  Monte-Carlo simulation of Moran's I
## data:  CRIME
## weights: colqueen
## number of simulations + 1: 1000
## statistic = 0.4858, observed rank = 1000, p-value =
## 0.001
## alternative hypothesis: greater
```

Moran's Test - Permutation Results

```
head(morpermCRIME$res)
```

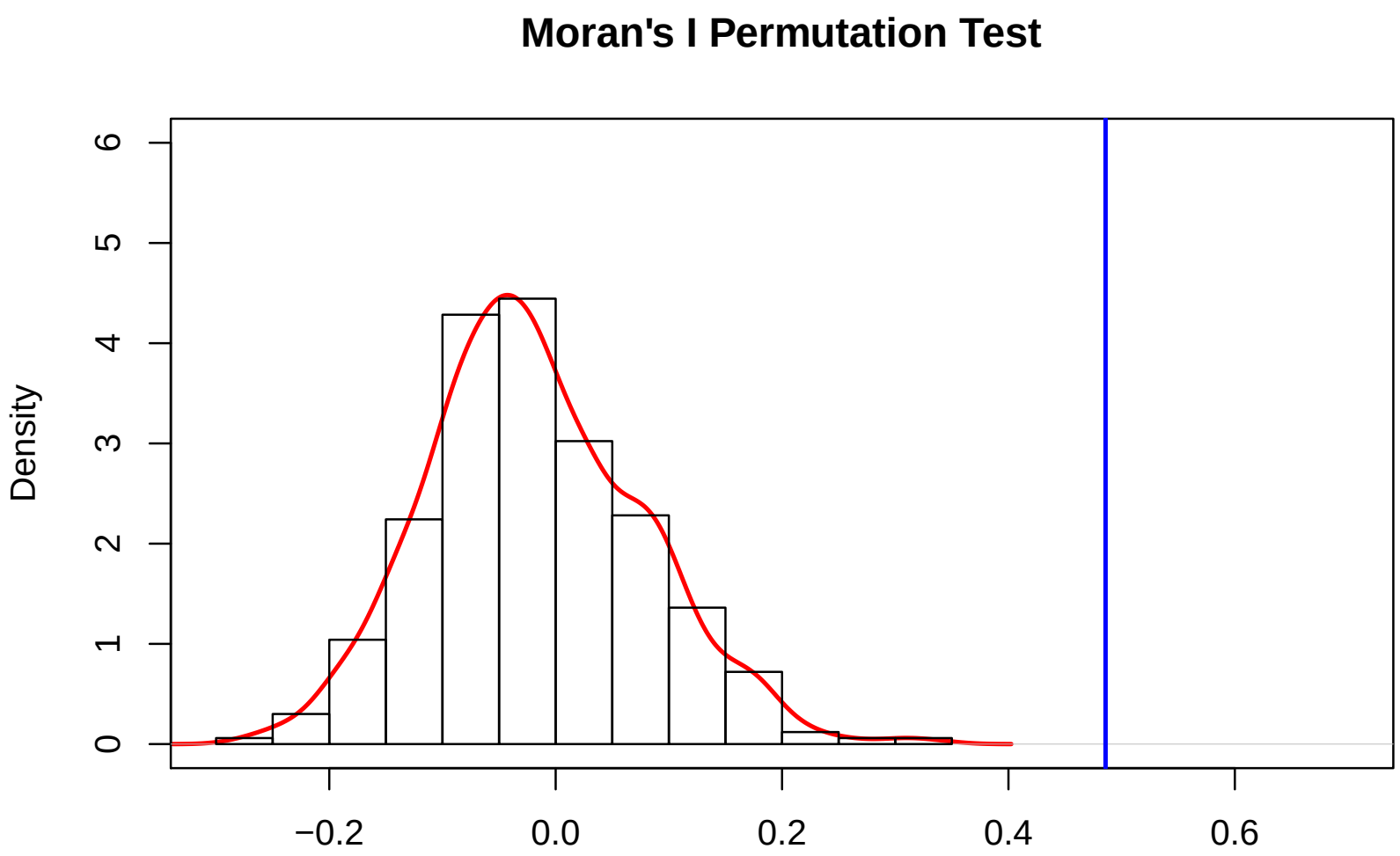
```
## [1] -0.06607790  0.01398150  0.07363572 -0.07023917
## [5] -0.15995950 -0.15462563
```

```
morp = morpermCRIME$res[1:length(morpermCRIME$res)-1]
```

density of the permutation results

```
plot(density(morp),main="Moran's I Permutation Test",
xlab="Reference Distribution",xlim=c(-0.3,0.7),
ylim=c(0,6),lwd=2,col="red")
hist(morp,freq=FALSE,add=TRUE)
abline(v=morpermCRIME$statistic,lwd=2,col="blue")
```

Moran's Test - Density of Permutaton Results



Construct Spatially Lagged Variable

```
xx = cbind(CRIME,INC,HOVAL)
wxx = lag.listw(colqueen,var=xx)
head(wxx)
```

```
##          [,1]     [,2]     [,3]
## [1,] 24.71427 18.59400 35.45850
## [2,] 26.24684 13.32133 46.67233
## [3,] 29.41175 14.12300 45.36475
## [4,] 34.64648 14.94425 32.81675
## [5,] 38.41199 11.81786 30.81786
## [6,] 40.62371 14.41900 37.91250
```

Calculate Spatially Lagged Variable - Manually

```
colqueen$neighbours[1]

## [[1]]
## [1] 2 3

colqueen$weights[1]

## [[1]]
## [1] 0.5 0.5

0.5*xx[2,] + 0.5*xx[3,]

##    CRIME       INC     HOVAL
## 24.71427 18.59400 35.45850
```

Calculate Spatially Lagged Variable - Matrix Multiplication

- access full weight matrix (not efficient) with **nb2mat**[3]

```
wmat = nb2mat(colqueen[[2]])
wmat[1,1:8]

## [1] 0.0 0.5 0.5 0.0 0.0 0.0 0.0 0.0

wxxmat = wmat %*% xx
wxx[1,1:3]

## [1] 24.71427 18.59400 35.45850

wxxmat[1,1:3]

##     CRIME       INC     HOVAL
## 24.71427 18.59400 35.45850
```

[3] The workbook shows an example with random numbers instead

Compare Variable with its Spatial Lag

- spatial lags lead to smaller range and variance of variables
- here: CRIME variable

```
summary(xx[,1])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.1783 20.0500 34.0000 35.1300 48.5900 68.8900
```

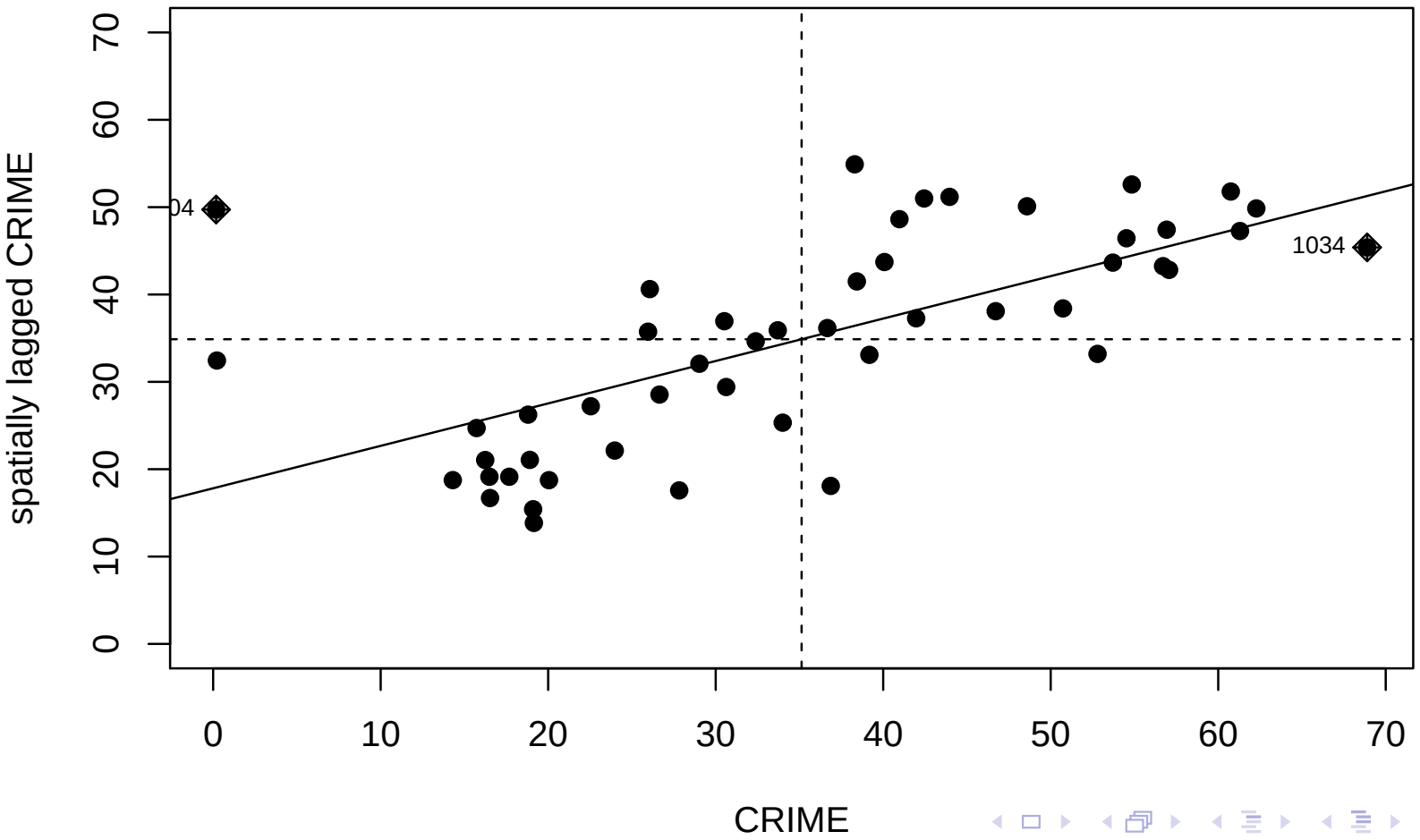
```
summary(wxx[,1])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   13.85   24.71   35.90   34.88   45.39   54.91
```

```
c(var(xx)[1],var(wxx)[1])
```

```
## [1] 279.9629 147.0571
```

Moran Plot



Moran Plot

```
moran.plot(CRIME,listw=colqueen,pch=19,ylim=c(0,70))
```

- smaller range of lagged variable visible
- slope of regression line equals Moran's I

Calculate Moran's I for CRIME variable

$$\hat{I} = \frac{z'Wz}{z'z}$$

```
z = CRIME - mean(CRIME)
zz = crossprod(z)
Wy = lag.listw(colqueen,z)
yWy = crossprod(z,Wy)
mi = yWy / zz
mi
```

```
##             [,1]
## [1,] 0.4857709
```

Calculate Properties of Moran's I

In the following, the properties of Moran's test with the CRIME variable are calculated[4]

$$E(\hat{I}) = \frac{-1}{n-1} \qquad \text{Var}(\hat{I}) = \frac{n^2 S_1 - nS_2 + 3S_0^2}{(n-1)(n+1)S_0^2} - \left(\frac{-1}{n-1}\right)^2$$

```
con = spweights.constants(colqueen)
t(con)
```

```
##       n  n1 n2 n3 nn   S0 S1       S2
## [1,] 49 48 47 46 2401 49 23.48489 204.6687
```

[4] The workbook shows an example with random numbers instead

Calculate Properties of Moran's I

```
E = -1/con$n1
E
```

```
## [1] -0.02083333
```

```
Esq = (con$n^2*con$S1-con$n*con$S2+3*con$S0^2) /
(con$n1*(con$n+1)*con$S0^2)
Esq
```

```
## [1] 0.00929499
```

```
VAR = Esq-E^2
VAR
```

```
## [1] 0.008860962
```

Calculate Test Statistic of Moran's Test

$$\hat{I}_z = \frac{\hat{I} - E(\hat{I})}{\sqrt{\text{Var}(\hat{I})}}$$

```
iz = (mi-E)/sqrt(VAR)
iz
```

```
##           [,1]
## [1,] 5.38181
```

p value

```
piz = pnorm(iz,lower.tail=FALSE)*2
piz
```

```
##                 [,1]
## [1,] 7.374047e-08
```

Moran's Test - Output

for comparison the output from above

```
moran.test(CRIME,colqueen,rand=FALSE,alt="two.sided")
```

```
##  Moran's I test under normality
## data:  CRIME
## weights: colqueen
## Moran I statistic standard deviate = 5.3818, p-value
## = 7.374e-08
## alternative hypothesis: two.sided
## sample estimates:
## Moran I statistic       Expectation          Variance
##      0.485770914     -0.020833333      0.008860962
```

Simulate Properties of Moran's I

- initialisation
  - set values of n, r (number of iterations)
  - create spatial weights object
- loop: r iterations
  - draw n random numbers (standard normal distributed)
  - calculate the properties of the Moran's test as above
- analysis
  - compare theoretical and empirical moments of Moran's I
  - plot empirical distribution of Moran's I
  - check frequency of H_0 rejection
  - result: the Moran's I test statistic *significantly under-rejects the null hypothesis* when n equals 16 using rook contiguity

SAR Model

$$\varepsilon = \rho W \varepsilon + u$$
$$\Leftrightarrow \varepsilon = (I - \rho W)^{-1} u$$

- **invIrM** constructs $(I - \rho W)^{-1}$

```
set.seed(123456)
SARinv = invIrM(colqueen[[2]],rho=0.5)
SARu = rnorm(49)
SAReps = SARinv %*% SARu
```

Moran's Test of SAR Variables

```
moran.test(SAReps,colqueen,rand=FALSE,alt="two.sided")

##  Moran's I test under normality
## data:  SAReps
## weights: colqueen
## Moran I statistic standard deviate = 6.0766, p-value
## = 1.228e-09
## alternative hypothesis: two.sided
## sample estimates:
## Moran I statistic       Expectation          Variance
##      0.551173601     -0.020833333      0.008860962
```

- spatial dependence

SMA Model

$$\varepsilon = \rho W u + u$$

```
set.seed(123456)
SMAu = rnorm(49)
SMAwu = lag.listw(colqueen,SMAu)
SMAeps = 0.5*SMAwu+SMAu
```

Moran's Test of SMA Variables

```
moran.test(SMAeps,colqueen,rand=FALSE,alt="two.sided")

##  Moran's I test under normality
## data:  SMAeps
## weights: colqueen
## Moran I statistic standard deviate = 5.0774, p-value
## = 3.826e-07
## alternative hypothesis: two.sided
## sample estimates:
## Moran I statistic       Expectation          Variance
##      0.457115534      -0.020833333       0.008860962
```

- spatial dependence
- greater Moran's I when considering SAR (c.p.)
- given the chosen random numbers the minimum Moran's I ($\rho = 0$) equals 0.175 (both models), the maximum ($\rho = 1$) equals 0.965 (SAR) and 0.614 respectively (SMA)

OLS

we consider OLS regression with spatial autoregressive errors

- estimators are still unbiased but less efficient
- standard errors are biased, thus test results cannot be interpreted

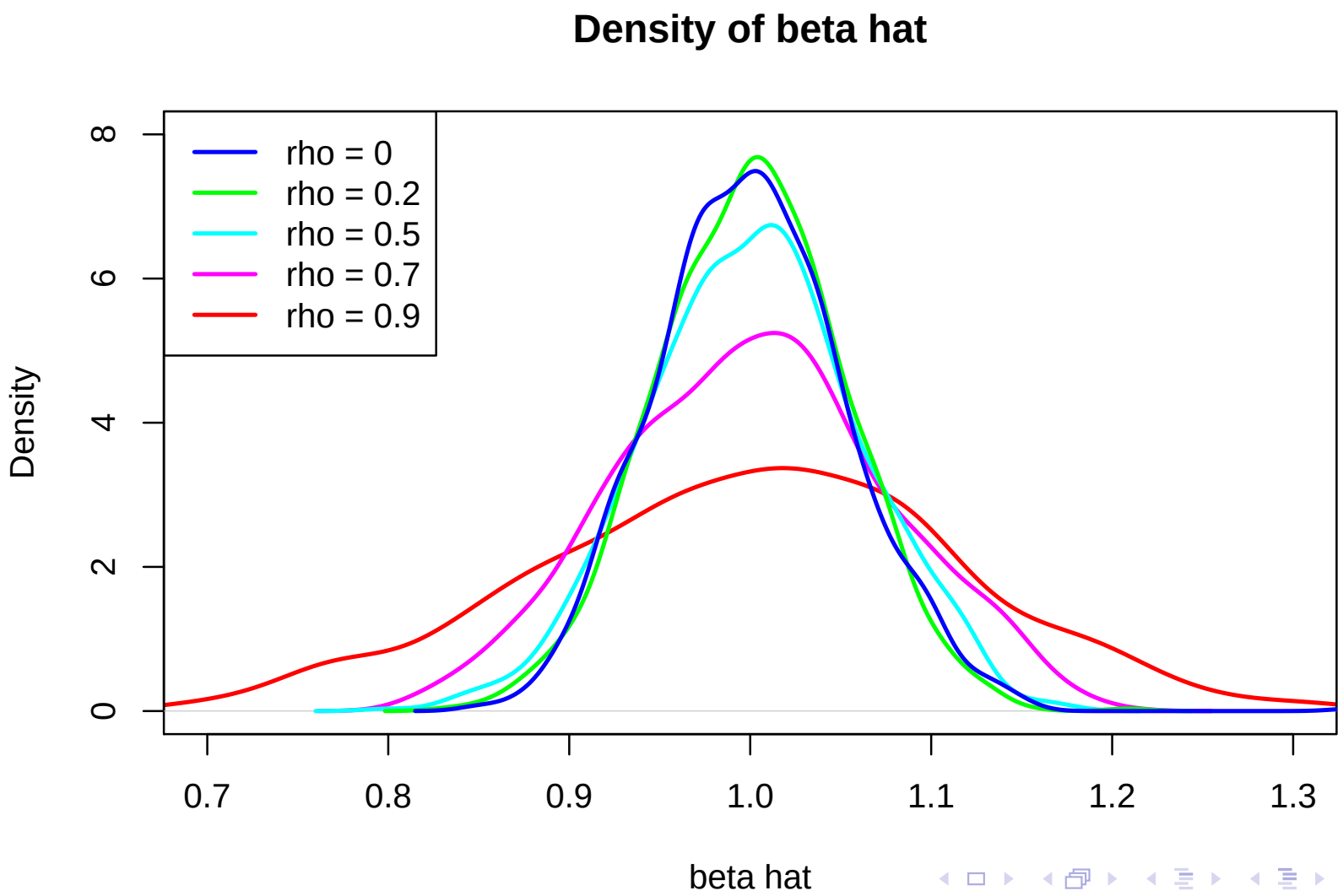
Simulate Properties of OLS Estimators 1/2

- initialisation
  - set values of n, r and vector with values of ρ
  - create X matrix as intended (here such that $R^2 = 0.9$ and $\beta = (1, 1)'$)
  - compute $(X'X)^{-1}$ and the variance of β (considered as a random variable)
  - create spatial weights object
- outer loop: 5 iterations for ρ
  - create SAR transformation matrices
- inner loop: r iterations
  - draw n random numbers (standard normal distribution)
  - create spatially correlated errors ε
  - compute dependent variable y
  - estimate β

Simulate Properties of OLS Estimators 2/2

- analysis
  - compare theoretical and empirical moments of β / $\hat{\beta}$
  - compute MSE of $\hat{\beta}$
  - plot empirical distribution of $\hat{\beta}$
- results
  - mean and median $\hat{\beta}$ are very close to β
  - larger ρ lead to larger range of $\hat{\beta}$, $\hat{\text{Var}}(\hat{\beta}|X)$ and MSE
  - their increase is greater the larger the increase of ρ

Simulation Results - Density Plot



Conclusion

- **spdep** offers different R commands for importing spatial file formats, creating and converting spatial objects, spatial data management, analysis and visualisation
- Moran's test tests for spatial dependence; its components could be replicated step by step
- simulations show the effects of spatial dependence on e.g. the OLS estimator

Spatial R Commands Used

```
library(spdep) # load spdep package
read.gal()     # import spatial weights file (.gal)
read.gwt2nb()  # import spatial weights file (.gwt)
cell2nb()      # create weights (neighbour list object)
nb2listw()     # convert neighbour list object to
               # spatial weights objects
nb2mat()       # convert neighbour list object to matrix
lag.listw()    # create spatially lagged variables
               # using a spatial weights object
invIrM()       # create inverse matrix (SAR context)
moran.test()   # Moran's test (randomis. or normal approx.)
moran.mc()     # Moran's test (simulation)
moran.plot()   # Moran plot
```

Further Useful Spatial R Commands

```
lagsarlm()      # Spatial Autoregressive Model
                # includes Spatial Durbin Model if needed
errorsarlm()    # Spatial Error Model
                # etype="emixed": Spatial Durbin Model
stsls()         # spatial models using
                # two stage least squares
GMerrorsar()    # spatial models using
                # generalised method of moments
lm.LMtests()    # LM test to test for 5 types of spatial dep.
```

Also have a look at the CRAN TaskView *Analysis of Spatial Data*[5] including further packages for additional purposes.

[5] http://cran.r-project.org/web/views/Spatial.html

References and Software Used


Anselin, Luc (2005). *Spatial Regression Analysis in R*. Workbook. University of Illinois - Spatial Analysis Laboratory. URL: https://geodacenter.asu.edu/system/files/rex1.pdf.


Anselin, Luc, Ibnu Syabri, and Youngihn Kho (2006). "GeoDa: An Introduction to Spatial Data Analysis". In: *Geographical Analysis, 2006, 38, 1, 5-22*. GeoDa version 1.6.6. ISSN: 0016-7363. URL: https://geodacenter.asu.edu/projects/opengeoda.


Bivand, Roger et al. (2015). *spdep: Spatial Dependence, Weighting Schemes, Statistics and Models*. R package version 0.5-82. URL: http://CRAN.R-project.org/package=spdep.


R Core Team (2014). *R: A Language and Environment for Statistical Computing*. R version 3.1.2. R Foundation for Statistical Computing. Vienna, Austria. URL: http://www.R-project.org/.


Xie, Yihui (2014). *knitr: A General-Purpose Package for Dynamic Report Generation in R*. R package version 1.8. URL: http://CRAN.R-project.org/package=knitr.

Thank you