ISTQL 2012 Summer School - Spatial Statistics and Econometrics

Şebnem Er, Neslihan Fidan

July 3-14, 2012

What is R?

- R was created by Ross Ihaka and Robert Gentleman at the University of Auckland.

What is R?

- R was created by Ross Ihaka and Robert Gentleman at the University of Auckland.
- R is named partly after the first names of the first two R authors (Robert Gentleman and Ross Ihaka), and partly as a play on the name of S, the famous statistical programming language.

What is R?

- R was created by Ross Ihaka and Robert Gentleman at the University of Auckland.
- R is named partly after the first names of the first two R authors (Robert Gentleman and Ross Ihaka), and partly as a play on the name of S, the famous statistical programming language.
- R is an integrated suite of software facilities for data manipulation and analysis, calculation and graphical display.

What is R?

- R was created by Ross Ihaka and Robert Gentleman at the University of Auckland.
- R is named partly after the first names of the first two R authors (Robert Gentleman and Ross Ihaka), and partly as a play on the name of S, the famous statistical programming language.
- R is an integrated suite of software facilities for data manipulation and analysis, calculation and graphical display.
- R is a free software.

What is R?

- R was created by Ross Ihaka and Robert Gentleman at the University of Auckland.
- R is named partly after the first names of the first two R authors (Robert Gentleman and Ross Ihaka), and partly as a play on the name of S, the famous statistical programming language.
- R is an integrated suite of software facilities for data manipulation and analysis, calculation and graphical display.
- R is a free software.
- R is very much a vehicle for newly developing methods of interactive data analysis. It has developed rapidly, and has been extended by a large collection of packages.

Using R

- Technically R is an expression language with a very simple syntax, and R uses a command line interface; however, several graphical user interfaces are available for use with R.

Using R

- Technically R is an expression language with a very simple syntax, and R uses a command line interface; however, several graphical user interfaces are available for use with R.
- R is CASE SENSITIVE as are most UNIX based packages, so A and a are different symbols and would refer to different variables.

Using R

- Technically R is an expression language with a very simple syntax, and R uses a command line interface; however, several graphical user interfaces are available for use with R.
- R is CASE SENSITIVE as are most UNIX based packages, so A and a are different symbols and would refer to different variables.
- R is also a calculator.

Using R

- Technically R is an expression language with a very simple syntax, and R uses a command line interface; however, several graphical user interfaces are available for use with R.
- R is CASE SENSITIVE as are most UNIX based packages, so A and a are different symbols and would refer to different variables.
- R is also a calculator.
- Elementary commands in R consist of either expressions or assignments. If an expression is given as a command, it is evaluated, printed (unless specifically made invisible), and the value is lost. An assignment also evaluates an expression and passes the value to a variable but the result is not automatically printed.

Using R

- Technically R is an expression language with a very simple syntax, and R uses a command line interface; however, several graphical user interfaces are available for use with R.
- R is CASE SENSITIVE as are most UNIX based packages, so A and a are different symbols and would refer to different variables.
- R is also a calculator.
- Elementary commands in R consist of either expressions or assignments. If an expression is given as a command, it is evaluated, printed (unless specifically made invisible), and the value is lost. An assignment also evaluates an expression and passes the value to a variable but the result is not automatically printed.
- R provides a mechanism for recalling and re-executing previous commands. The vertical arrow keys on the keyboard can be used to scroll forward and backward through a command history.

Executing Commands from an External File

You can either use the R Console or write your scripts in an R file.
That way you can keep your codes and run them the next time you open R.

```
> source("week1day1.R")
```

Object Oriented R

The entities that R creates and manipulates are known as objects. These may be variables, arrays of numbers, character strings, functions, or more general structures built from such components. During an R session, objects are created and stored by name. The R command:

```
> objects()
```

Alternatively, ls() can be used to display the names of (most of) the objects which are currently stored within R. The collection of objects currently stored is called the workspace.

Object Oriented R

All objects created during an R session can be stored permanently in a file for use in future R sessions. At the end of each R session you are given the opportunity to save all the currently available objects. If you indicate that you want to do this, the objects are written to a file called '.Rdata' in the current directory, and the command lines used in the session are saved to a file called '.Rhistory'.

Object Oriented R

All objects created during an R session can be stored permanently in a file for use in future R sessions. At the end of each R session you are given the opportunity to save all the currently available objects. If you indicate that you want to do this, the objects are written to a file called '.Rdata' in the current directory, and the command lines used in the session are saved to a file called '.Rhistory'.

When R is started at later time from the same directory it reloads the workspace from this file. At the same time the associated commands history is reloaded. It is recommended that you should use separate working directories for analyses conducted with R. It is quite common for objects with names x and y to be created during an analysis.

How to Set your Working Directory

Workspace: All objects created in R are stored in workspace (using backward slash twice)

How to Set your Working Directory

Workspace: All objects created in R are stored in workspace (using backward slash twice)

```
> setwd("C:\\Users\\TOSHIBA\\Downloads\\Spatial\\AnilBERA\\latexpdf")
```

How to Set your Working Directory

Workspace: All objects created in R are stored in workspace (using backward slash twice)

```
> setwd("C:\\Users\\TOSHIBA\\Downloads\\Spatial\\AnilBERA\\latexpdf")
```

or equivalently (using forward slash only for once)

How to Set your Working Directory

Workspace: All objects created in R are stored in workspace (using backward slash twice)

```
> setwd("C:\\Users\\TOSHIBA\\Downloads\\Spatial\\AnilBERA\\latexpdf")
```

or equivalently (using forward slash only for once)

```
> setwd("C:/Users/TOSHIBA/Downloads/Spatial/AnilBERA/latexpdf")
```

How to Set your Working Directory

Workspace: All objects created in R are stored in workspace (using backward slash twice)

```
> setwd("C:\\Users\\TOSHIBA\\Downloads\\Spatial\\AnilBERA\\latexpdf")
```

or equivalently (using forward slash only for once)

```
> setwd("C:/Users/TOSHIBA/Downloads/Spatial/AnilBERA/latexpdf")
```

in order to see which directory is assigned:

```
> getwd()
[1] "C:/Users/TOSHIBA/Downloads/Spatial/AnilBERA/latexpdf"
```

Getting Help

```
> help()
> help.start()
> help(mean)
> ?mean
> demo()
> ?read.table
> help.search ("data.entry")
> apropos (boxplot)
```

Vectors and Assignment

R operates on named data structures. The simplest such structure is the numeric vector, which is a single entity consisting of an ordered collection of numbers. To set up a vector named y, say, consisting of five numbers, namely 10.4, 5.6, 3.1, 6.4 and 21.7, use the R command:

```
> y <- c(10.4, 5.6, 3.1, 6.4, 21.7)
```

Vectors and Assignment

R operates on named data structures. The simplest such structure is the numeric vector, which is a single entity consisting of an ordered collection of numbers. To set up a vector named y, say, consisting of five numbers, namely 10.4, 5.6, 3.1, 6.4 and 21.7, use the R command:

```
> y <- c(10.4, 5.6, 3.1, 6.4, 21.7)
```

This is an assignment statement using the function c() which in this context can take an arbitrary number of vector arguments and whose value is a vector got by concatenating its arguments end to end. A number occurring by itself in an expression is taken as a vector of length one.

Vectors and Assignment

Notice that the assignment operator ('<-'), which consists of the two characters '<' ("less than") and '-' ("minus") occurring strictly side-by-side and it 'points' to the object receiving the value of the expression. In most contexts the '=' operator can be used as an alternative.

```
> y = c(10.4, 5.6, 3.1, 6.4, 21.7)
```

Vectors and Assignment

Notice that the assignment operator ('<-'), which consists of the two characters '<' ("less than") and '-' ("minus") occurring strictly side-by-side and it 'points' to the object receiving the value of the expression. In most contexts the '=' operator can be used as an alternative.

```
> y = c(10.4, 5.6, 3.1, 6.4, 21.7)

> x <- 1:10
> y <- x + rnorm(10,0,1)
> x+y

 [1]  3.079802  3.065141  6.812409  7.005371 10.215046 11.775668
 [7] 13.738748 17.274728 18.373461 18.107064
```

Writing a Sequence

```
> x = 1:4
> x
[1] 1 2 3 4
```

Writing a Sequence

```
> x = 1:4
> x
[1] 1 2 3 4
> xsquare = x**2
> xsquare
[1]  1  4  9 16
```

Writing a Sequence

```
> x = 1:4
> x
[1] 1 2 3 4
> xsquare = x**2
> xsquare
[1]  1  4  9 16
> prod1 = x*10
> prod1
[1] 10 20 30 40
```

Writing a Sequence

The function seq() is a more general facility for generating sequences. It has five arguments, only some of which may be specified in any one call. The first two arguments, if given, specify the beginning and end of the sequence, and if these are the only two arguments given the result is the same as the colon operator.

```
> seq(-5, 5, by=.2) -> s3
```

generates in s3 the vector c(-5.0, -4.8, -4.6, ..., 4.6, 4.8, 5.0).

Writing a Sequence

The function seq() is a more general facility for generating sequences. It has five arguments, only some of which may be specified in any one call. The first two arguments, if given, specify the beginning and end of the sequence, and if these are the only two arguments given the result is the same as the colon operator.

```
> seq(-5, 5, by=.2) -> s3
```

generates in s3 the vector c(-5.0, -4.8, -4.6, ..., 4.6, 4.8, 5.0).
Similarly,

Writing a Sequence

The function seq() is a more general facility for generating sequences. It has five arguments, only some of which may be specified in any one call. The first two arguments, if given, specify the beginning and end of the sequence, and if these are the only two arguments given the result is the same as the colon operator.

```
> seq(-5, 5, by=.2) -> s3
```

generates in s3 the vector c(-5.0, -4.8, -4.6, ..., 4.6, 4.8, 5.0).
Similarly,

```
> s4 <- seq(length=51, from=-5, by=.2)
```

generates the same vector in s4.

More on Vectors

Remember the vector y?

```
> y = c(10.4, 5.6, 3.1, 6.4, 21.7)
```

More on Vectors

Remember the vector y?

```
> y = c(10.4, 5.6, 3.1, 6.4, 21.7)
```

Let's say you need to know the length of the vector to write a for loop.

```
> length(y)
[1] 5
```

More on Vectors

Remember the vector y?

```
> y = c(10.4, 5.6, 3.1, 6.4, 21.7)
```

Let's say you need to know the length of the vector to write a for loop.

```
> length(y)
```

```
[1] 5
```

To concatenate vectors

```
> z<-cbind(x,y) # combines x and y by columns
> v<-rbind(x,y) # by rows
```

More on Vectors

Remember the vector y?

```
> y = c(10.4, 5.6, 3.1, 6.4, 21.7)
```

Let's say you need to know the length of the vector to write a for loop.

```
> length(y)
[1] 5
```

To concatenate vectors

```
> z<-cbind(x,y) # combines x and y by columns
> v<-rbind(x,y) # by rows
```

To learn the dimensions

```
> dim(z) # 3 rows 2 columns
[1] 5 2
> dim(v)
[1] 2 5
> length(z) # number of elements
[1] 10
```

Vector arithmetic

Vectors can be used in arithmetic expressions, in which case the operations are performed element by element. The elementary arithmetic operators are the usual $+$, $-$, $*$, $/$ and a ^ for raising to a power. In addition all of the common arithmetic functions are available. log, exp, sin, cos, tan, sqrt, and so on, all have their usual meaning:

Vector arithmetic

Vectors can be used in arithmetic expressions, in which case the operations are performed element by element. The elementary arithmetic operators are the usual $+$, $-$, $*$, $/$ and a ^ for raising to a power. In addition all of the common arithmetic functions are available. log, exp, sin, cos, tan, sqrt, and so on, all have their usual meaning:

```
> downtime =c(0, 1, 2, 12, 12, 14, 18, 21, 21, 23,
+ 24,25,28,29,30,30,30,33,36,44,45,47,51)
> mean(downtime)

[1] 25.04348

> median(downtime)

[1] 25

> range(downtime)

[1]  0 51

> sd(downtime)

[1] 14.27164
```

Vector arithmetic

```
> length(downtime)
[1] 23
> log(downtime)
 [1]      -Inf 0.0000000 0.6931472 2.4849066 2.4849066 2.6390573
 [7] 2.8903718 3.0445224 3.0445224 3.1354942 3.1780538 3.2188758
[13] 3.3322045 3.3672958 3.4011974 3.4011974 3.4011974 3.4965076
[19] 3.5835189 3.7841896 3.8066625 3.8501476 3.9318256
> max(downtime)
[1] 51
> min(downtime)
[1] 0
> sum(downtime)
[1] 576
> prod(downtime)
[1] 0
```

Other types of objects

Vectors are the most important type of object in R, but there are several others.

- matrices or more generally arrays are multi-dimensional generalizations of vectors. In fact, they are vectors that can be indexed by two or more indices and will be printed in special ways.

Other types of objects

Vectors are the most important type of object in R, but there are several others.

- matrices or more generally arrays are multi-dimensional generalizations of vectors. In fact, they are vectors that can be indexed by two or more indices and will be printed in special ways.
- factors provide compact ways to handle categorical data.

Other types of objects

Vectors are the most important type of object in R, but there are several others.

- matrices or more generally arrays are multi-dimensional generalizations of vectors. In fact, they are vectors that can be indexed by two or more indices and will be printed in special ways.
- factors provide compact ways to handle categorical data.
- lists are a general form of vector in which the various elements need not be of the same type, and are often themselves vectors or lists. Lists provide a convenient way to return the results of a statistical computation.

Other types of objects

Vectors are the most important type of object in R, but there are several others.

- matrices or more generally arrays are multi-dimensional generalizations of vectors. In fact, they are vectors that can be indexed by two or more indices and will be printed in special ways.
- factors provide compact ways to handle categorical data.
- lists are a general form of vector in which the various elements need not be of the same type, and are often themselves vectors or lists. Lists provide a convenient way to return the results of a statistical computation.
- data frames are matrix-like structures, in which the columns can be of different types. Think of data frames as 'data matrices' with one row per observational unit but with (possibly) both numerical and categorical variables. Many experiments are best described by data frames: the treatments are categorical but the response is numeric.

Other types of objects

Vectors are the most important type of object in R, but there are several others.

- matrices or more generally arrays are multi-dimensional generalizations of vectors. In fact, they are vectors that can be indexed by two or more indices and will be printed in special ways.
- factors provide compact ways to handle categorical data.
- lists are a general form of vector in which the various elements need not be of the same type, and are often themselves vectors or lists. Lists provide a convenient way to return the results of a statistical computation.
- data frames are matrix-like structures, in which the columns can be of different types. Think of data frames as 'data matrices' with one row per observational unit but with (possibly) both numerical and categorical variables. Many experiments are best described by data frames: the treatments are categorical but the response is numeric.
- functions are themselves objects in R which can be stored in the project's workspace.

Create a New Matrix

A matrix is the special case of a two-dimensional array with the following syntax:

```
matrix(data,nrow,ncol,byrow)
```

Create a New Matrix

A matrix is the special case of a two-dimensional array with the following syntax:

```
matrix(data,nrow,ncol,byrow)
```

Let's start with a vector of size 12:

```
> a<-c(2,-2,9,3,1,4,5,4,0,1,4,-3)
```

Create a New Matrix

A matrix is the special case of a two-dimensional array with the following syntax:

```
matrix(data,nrow,ncol,byrow)
```

Let's start with a vector of size 12:

```
> a<-c(2,-2,9,3,1,4,5,4,0,1,4,-3)
```

If you want to transform this vector into a 3X4 matrix:

```
> A<- matrix(a, nrow=3,ncol=4) # creates a matrix based on columns (default).
> A
```

```
     [,1] [,2] [,3] [,4]
[1,]    2    3    5    1
[2,]   -2    1    4    4
[3,]    9    4    0   -3
```

If you want you can create a matrix by rows with the byrow=TRUE option.

```
> length(a) # dim(a)-> NULL
[1] 12
> dim(A)
[1] 3 4
> nrow(A)
[1] 3
> ncol(A)
[1] 4
> length(A) # number of the elements in A matrix
[1] 12
```

```
> length(a) # dim(a)-> NULL
[1] 12
> dim(A)
[1] 3 4
> nrow(A)
[1] 3
> ncol(A)
[1] 4
> length(A) # number of the elements in A matrix
[1] 12
```

If you will need to work with the elements of a matrix

```
> A[3,]
[1]  9  4  0 -3
```

```
> A[3,2]
[1] 4
```

```
> length(a) # dim(a)-> NULL
[1] 12
> dim(A)
[1] 3 4
> nrow(A)
[1] 3
> ncol(A)
[1] 4
> length(A) # number of the elements in A matrix
[1] 12
```

If you will need to work with the elements of a matrix

```
> A[3,]
[1]  9  4  0 -3
> A[3,2]
[1] 4
> A[,3]
[1] 5 4 0
> A[1]
[1] 2
> A[1:3,2] # 1,2,3rd rows on second column.
[1] 3 1 4
> A[c(1,3),] # first and third rows and all the
     [,1] [,2] [,3] [,4]
[1,]    2    3    5    1
[2,]    9    4    0   -3
> A[c(1,3),2]
[1] 3 4
```

Diagonal Matrix and Type of Objects

To obtain the diagonal elements of a matrix

```
> diag(A)
[1] 2 1 0
```

Diagonal Matrix and Type of Objects

To obtain the diagonal elements of a matrix

```
> diag(A)
[1] 2 1 0
```

To learn the object's type use class function:

```
> class(A)  # matrix
[1] "matrix"
> class(a)  # numeric (vector)
[1] "numeric"
> class(A[1,]) # numeric (vector)
[1] "numeric"
```

Unit Matrix

How do you think you would generate a Unit Matrix?

Unit Matrix

How do you think you would generate a Unit Matrix?

```
> I<-diag(1,nrow=3,ncol=3) # 3-by-3
> zeros <- matrix(0,nrow=3,ncol=3) # all the elements equal to zero
> ones <- matrix(1,nrow=3,ncol=3) # all the elements equal to 1
> I<- matrix(0,nrow=3,ncol=3) #preparing unit matris
> I[row(I)==col(I)] <- 1 # diagonal elements equal 1
> I

     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1
```

Matrix Operations

Let's create another matrix called B:

```
> b<-c(0,-1,-4,0,3,4,7,2,-3,10,4,8)
> B<- matrix(b, nrow=3,ncol=4)
> B
```

```
     [,1] [,2] [,3] [,4]
[1,]    0    0    7   10
[2,]   -1    3    2    4
[3,]   -4    4   -3    8
```

Matrix Operations

Let's create another matrix called B:

```
> b<-c(0,-1,-4,0,3,4,7,2,-3,10,4,8)
> B<- matrix(b, nrow=3,ncol=4)
> B
```

```
     [,1] [,2] [,3] [,4]
[1,]    0    0    7   10
[2,]   -1    3    2    4
[3,]   -4    4   -3    8
```

Summation/subtraction of two matrices:

```
> C<-A+B # remember A and B should have same dimension
> Y<-A-B
```

Matrix Operations

Let's create another matrix called B:

```
> b<-c(0,-1,-4,0,3,4,7,2,-3,10,4,8)
> B<- matrix(b, nrow=3,ncol=4)
> B
```

```
     [,1] [,2] [,3] [,4]
[1,]    0    0    7   10
[2,]   -1    3    2    4
[3,]   -4    4   -3    8
```

Summation/subtraction of two matrices:

```
> C<-A+B # remember A and B should have same dimension
> Y<-A-B
```

Using a scalar:

```
> C5<-5*A # (or C5<-A*5 same)
> Cp5<-A+3
```

Product of two matrices (mxn & nxk)

```
> dim(A)
[1] 3 4
> dim(B)
[1] 3 4
```

Product of two matrices (mxn & nxk)

```
> dim(A)
```

```
[1] 3 4
```

```
> dim(B)
```

```
[1] 3 4
```

Since both A and B are 3X4, we need the transpose of A or B to be able to multiply A and B.

```
> t(A) # transpose of matrix A
```

```
     [,1] [,2] [,3]
[1,]    2   -2    9
[2,]    3    1    4
[3,]    5    4    0
[4,]    1    4   -3
```

```
> dim(t(A))
```

```
[1] 4 3
```

```
> P<-A%*%t(B) # (3x3)
> P4<-t(A)%*%B # (4x4)
```

Inverse of a Matrix - Square Matrix

```
> D<-matrix(c(2,4,5,3,-1,7,-3,5,-2),nrow=3,ncol=3)
> Di<-solve(D)
> round(Di%*%D) # should yield unit matrix

     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1

> Di

     [,1]        [,2]       [,3]
[1,]  0.5  0.22727273 -0.1818182
[2,] -0.5 -0.16666667  0.3333333
[3,] -0.5 -0.01515152  0.2121212
```

Cholesky Decomposition

Knowing that

- A is square matrix and positive definite
- A=LL* L, lower triangular

The Cholesky decomposition is mainly used for the numerical solution of linear equations Ax = b. If A is symmetric and positive definite, then we can solve $Ax = b$ by first computing the Cholesky decomposition $A = LL'$, then solving $Ly = b$ for y, and finally solving $L'x = y$ for x.

```
> T=matrix(c(1,1,1,1,5,5,1,5,14),nrow=3)
> chol(T)

     [,1] [,2] [,3]
[1,]    1    1    1
[2,]    0    2    2
[3,]    0    0    3
```

Lists and Data Frames

Data Entry: Entering data from a file to a data frame

Lists and Data Frames

Data Entry: Entering data from a file to a data frame

```
> boston.c =read.table("C:/data/bostondata.csv",header=T,sep=",")
```

Lists and Data Frames

Data Entry: Entering data from a file to a data frame

```
> boston.c =read.table("C:/data/bostondata.csv",header=T,sep=",")
```

if you don't want to write the directory each time you can specify a folder in a specific directory and assign this directory as your work place, remember setwd()?

Lists and Data Frames

Data Entry: Entering data from a file to a data frame

```
> boston.c =read.table("C:/data/bostondata.csv",header=T,sep=",")
```

if you don't want to write the directory each time you can specify a folder in a specific directory and assign this directory as your work place, remember setwd()?

```
> boston.c=read.table("bostondata.csv",header=T,sep=",")
```

The file name must be always written in quotes:

Lists and Data Frames

Data Entry: Entering data from a file to a data frame

```
> boston.c =read.table("C:/data/bostondata.csv",header=T,sep=",")
```

if you don't want to write the directory each time you can specify a folder in a specific directory and assign this directory as your work place, remember setwd()?

```
> boston.c=read.table("bostondata.csv",header=T,sep=",")
```

The file name must be always written in quotes:

```
> names(boston.c)
```

```
 [1] "X"       "TOWN"    "TOWNNO"  "TRACT"   "LON"     "LAT"
 [7] "MEDV"    "CMEDV"   "CRIM"    "ZN"      "INDUS"   "CHAS"
[13] "NOX"     "RM"      "AGE"     "DIS"     "RAD"     "TAX"
[19] "PTRATIO" "B"       "LSTAT"
```

Lists and Data Frames

Data Entry: Entering data from a file to a data frame

```
> boston.c =read.table("C:/data/bostondata.csv",header=T,sep=",")
```

if you don't want to write the directory each time you can specify a folder in a specific directory and assign this directory as your work place, remember setwd()?

```
> boston.c=read.table("bostondata.csv",header=T,sep=",")
```

The file name must be always written in quotes:

```
> names(boston.c)
```

```
 [1] "X"       "TOWN"    "TOWNNO"  "TRACT"   "LON"     "LAT"
 [7] "MEDV"    "CMEDV"   "CRIM"    "ZN"      "INDUS"   "CHAS"
[13] "NOX"     "RM"      "AGE"     "DIS"     "RAD"     "TAX"
[19] "PTRATIO" "B"       "LSTAT"
```

If you want to see the value of a specific observation, then type:

```
> boston.c$CRIM[4]
```

```
[1] 0.03237
```

Lists and Data Frames

Missing Values

```
> mean(boston.c$CRIM)
[1] 3.613524
```

Lists and Data Frames

Missing Values

```
> mean(boston.c$CRIM)
[1] 3.613524
```

If no result, that could be because of some data missing.

Lists and Data Frames

Missing Values

> *mean(boston.c$CRIM)*

```
[1] 3.613524
```

If no result, that could be because of some data missing.

`na.rm = T (not available, remove)`

or

`na.rm = TRUE`

Lists and Data Frames

You can specify the columns of the variables that you can actually get the mean for.

```
> mean(boston.c[,7:9])
     MEDV     CMEDV      CRIM
22.532806 22.528854  3.613524
```

Lists and Data Frames

You can specify the columns of the variables that you can actually get the mean for.

```
> mean(boston.c[,7:9])
     MEDV     CMEDV      CRIM
22.532806 22.528854  3.613524
> median(boston.c[,8])
[1] 21.2
```

Lists and Data Frames

```
> rangeCrime = max(boston.c$CRIM, na.rm=T)-min(boston.c$CRIM, na.rm=T)
> rangeCrime
[1] 88.96988
```

or equivalently,

```
> range(boston.c$CRIM,na.rm=T)
[1]  0.00632 88.97620
```

Lists and Data Frames

```
> rangeCrime = max(boston.c$CRIM, na.rm=T)-min(boston.c$CRIM, na.rm=T)
> rangeCrime
```

```
[1] 88.96988
```

or equivalently,

```
> range(boston.c$CRIM,na.rm=T)
```

```
[1]  0.00632 88.97620
```

```
> sd(boston.c$CRIM, na.rm=T)
```

```
[1] 8.601545
```

Lists and Data Frames

For the quantiles

```
> quantile(boston.c$CRIM, na.rm=T)
```

```
      0%       25%       50%       75%      100%
0.006320  0.082045  0.256510  3.677083 88.976200
```

Lists and Data Frames

For the quantiles

```
> quantile(boston.c$CRIM, na.rm=T)

       0%       25%       50%       75%      100%
 0.006320  0.082045  0.256510  3.677083 88.976200
```

For the Deciles

```
> deciles <-seq(0,1,0.1)
> deciles

 [1] 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0

> quantile(boston.c$CRIM, deciles,na.rm=T)

       0%       10%       20%       30%       40%       50%
 0.006320  0.038195  0.064170  0.099245  0.150380  0.256510
      60%       70%       80%       90%      100%
 0.550070  1.728440  5.581070 10.753000 88.976200
```

Lists and Data Frames

For the percentiles

```
> percentiles <-seq(0,1,0.01)
> perc100BostonCrime=quantile(boston.c$CRIM, percentiles,na.rm=T)
> perc100BostonCrime[1:21]
```

```
       0%        1%        2%        3%        4%        5%
0.0063200 0.0136105 0.0150470 0.0195310 0.0224920 0.0279100
       6%        7%        8%        9%       10%       11%
0.0306820 0.0338280 0.0352080 0.0359795 0.0381950 0.0416250
      12%       13%       14%       15%       16%       17%
0.0436220 0.0455440 0.0479560 0.0507700 0.0541440 0.0559585
      18%       19%       20%
0.0577550 0.0612890 0.0641700
```

Lists and Data Frames

```
> summary(boston.c,na.rm=T)
       X                        TOWN         TOWNNO
 Min.   :  1.0   Cambridge        : 30   Min.   : 0.00
 1st Qu.:127.2   Boston Savin Hill: 23   1st Qu.:26.25
 Median :253.5   Lynn             : 22   Median :42.00
 Mean   :253.5   Boston Roxbury   : 19   Mean   :47.53
 3rd Qu.:379.8   Newton           : 18   3rd Qu.:78.00
 Max.   :506.0   Somerville       : 15   Max.   :91.00
                 (Other)          :379
     TRACT            LON              LAT             MEDV
 Min.   :   1   Min.   :-71.29   Min.   :42.03   Min.   : 5.00
 1st Qu.:1303   1st Qu.:-71.09   1st Qu.:42.18   1st Qu.:17.02
 Median :3394   Median :-71.05   Median :42.22   Median :21.20
 Mean   :2700   Mean   :-71.06   Mean   :42.22   Mean   :22.53
 3rd Qu.:3740   3rd Qu.:-71.02   3rd Qu.:42.25   3rd Qu.:25.00
 Max.   :5082   Max.   :-70.81   Max.   :42.38   Max.   :50.00

     CMEDV            CRIM               ZN
 Min.   : 5.00   Min.   : 0.00632   Min.   :  0.00
 1st Qu.:17.02   1st Qu.: 0.08204   1st Qu.:  0.00
 Median :21.20   Median : 0.25651   Median :  0.00
 Mean   :22.53   Mean   : 3.61352   Mean   : 11.36
 3rd Qu.:25.00   3rd Qu.: 3.67708   3rd Qu.: 12.50
 Max.   :50.00   Max.   :88.97620   Max.   :100.00

     INDUS            CHAS              NOX
 Min.   : 0.46   Min.   :0.00000   Min.   :0.3850
 1st Qu.: 5.19   1st Qu.:0.00000   1st Qu.:0.4490
 Median : 9.69   Median :0.00000   Median :0.5380
 Mean   :11.14   Mean   :0.06917   Mean   :0.5547
 3rd Qu.:18.10   3rd Qu.:0.00000   3rd Qu.:0.6240
 Max.   :27.74   Max.   :1.00000   Max.   :0.8710

       RM             AGE              DIS
 Min.   :3.561   Min.   :  2.90   Min.   : 1.130
 1st Qu.:5.886   1st Qu.: 45.02   1st Qu.: 2.100
 Median :6.208   Median : 77.50   Median : 3.207
 Mean   :6.285   Mean   : 68.57   Mean   : 3.795
 3rd Qu.:6.623   3rd Qu.: 94.08   3rd Qu.: 5.188
 Max.   :8.780   Max.   :100.00   Max.   :12.127

      RAD              TAX           PTRATIO            B
 Min.   : 1.000   Min.   :187.0   Min.   :12.60   Min.   :  0.32
 1st Qu.: 4.000   1st Qu.:279.0   1st Qu.:17.40   1st Qu.:375.38
 Median : 5.000   Median :330.0   Median :19.05   Median :391.44
 Mean   : 9.549   Mean   :408.2   Mean   :18.46   Mean   :356.67
 3rd Qu.:24.000   3rd Qu.:666.0   3rd Qu.:20.20   3rd Qu.:396.23
 Max.   :24.000   Max.   :711.0   Max.   :22.00   Max.   :396.90

     LSTAT
 Min.   : 1.73
 1st Qu.: 6.95
 Median :11.36
 Mean   :12.65
 3rd Qu.:16.95
 Max.   :37.97
```

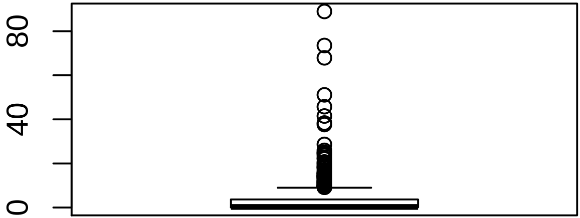
Lists and Data Frames

```
> summary(boston.c$CRIM,na.rm=T)

    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
 0.00632  0.08204  0.25650  3.61400  3.67700 88.98000
```

Boxplots

```
> boxplot(boston.c$CRIM)
```

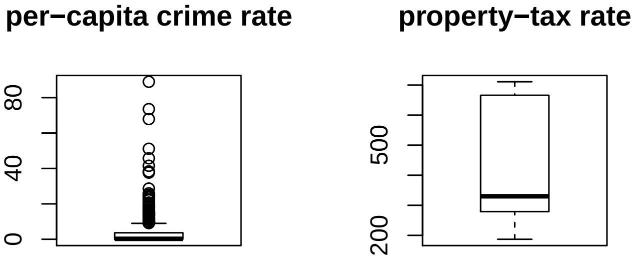


Boxplots

```
> boxplot(boston.c$CRIM,xlab = "Crime Rate",ylab = "(%)",cex.lab=0.5)
```

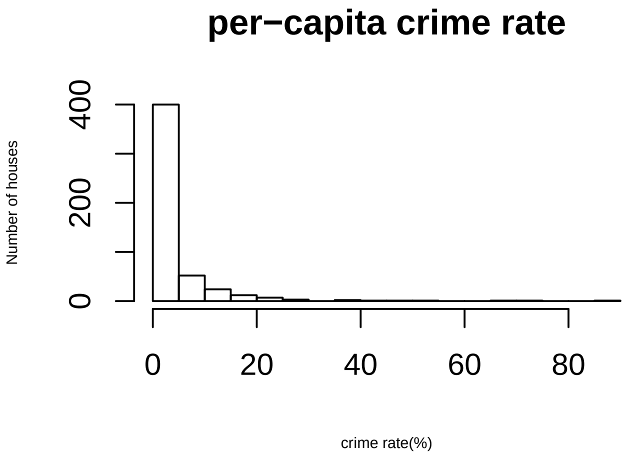
Multiple Boxplots

```
> par (mfrow = c(1,2))
> boxplot (boston.c$CRIM, main = "per-capita crime rate")
> boxplot(boston.c$TAX, main = "property-tax rate")
```



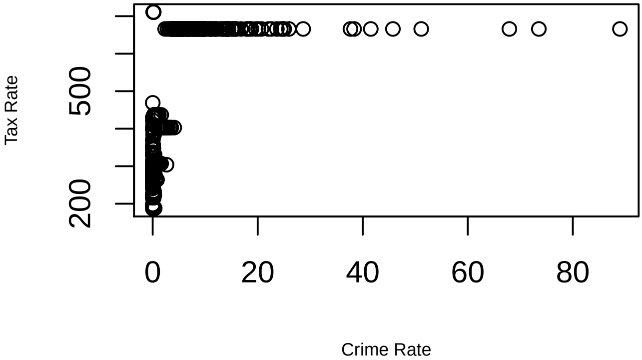
Histograms

```
> hist(boston.c$CRIM, breaks = 15,xlab ="crime rate(%)",ylab =
+ "Number of houses",main = "per-capita crime rate",cex.lab=0.5)
```



Scatter Plots

```
> plot(boston.c$CRIM,boston.c$TAX,xlab = "Crime Rate",
+ ylab = "Tax Rate",cex.lab=0.6)
```



What is a Spatial Data

All spatial data consist of positional information, answering the question 'where is it?'. (Bivand, et all., 2008, pp.8)

What is a Spatial Data

All spatial data consist of positional information, answering the question 'where is it?'. (Bivand, et all., 2008, pp.8)
The different types of data models that are distinguished in R include the following: (Bivand, et all., 2008, pp.8)

What is a Spatial Data

All spatial data consist of positional information, answering the question 'where is it?'. (Bivand, et all., 2008, pp.8)
The different types of data models that are distinguished in R include the following: (Bivand, et all., 2008, pp.8)

1. Continuous data (surfaces)
2. Discrete data
   - Point, a single point location, such as a GPS reading or a geocoded address
   - Line, a set of ordered points, connected by straight line segments
   - Polygon (regular-irregular), an area, marked by one or more enclosing lines, possibly containing holes. A polygon is formed when a set of line segments forms a closed object with no lines intersecting.
   - Grid, a collection of points or rectangular cells, organised in a regular lattice

What is a Spatial Data

All spatial data consist of positional information, answering the question ‘where is it?’. (Bivand, et all., 2008, pp.8)
The different types of data models that are distinguished in R include the following: (Bivand, et all., 2008, pp.8)

1. Continuous data (surfaces)
2. Discrete data
    - Point, a single point location, such as a GPS reading or a geocoded address
    - Line, a set of ordered points, connected by straight line segments
    - Polygon (regular-irregular), an area, marked by one or more enclosing lines, possibly containing holes. A polygon is formed when a set of line segments forms a closed object with no lines intersecting.
    - Grid, a collection of points or rectangular cells, organised in a regular lattice

Spatial data are usually displayed on maps, where the x- and y-axes show the coordinate values, with the aspect ratio chosen such that a unit in x equals a unit in y. (Bivand, et all., 2008)

R and Spatial Objects

Every object in R is of a particular class. You have seen numeric, vector, matrix and data.frame classes. To learn the type of the object type in R:

```
> x=c(1,2,3,5)
> class(x)
[1] "numeric"
```

R and Spatial Objects

Every object in R is of a particular class. You have seen numeric, vector, matrix and data.frame classes. To learn the type of the object type in R:

```
> x=c(1,2,3,5)
> class(x)
[1] "numeric"
```

In R, for spatial data, the foundation class is the Spatial class, with just two slots. The first is a bounding box, a matrix of numerical coordinates with column names c(min,max), and at least two rows, with the first row eastings (x-axis) and the second northings (y-axis). The second is a CRS class object defining the coordinate reference system. (Bivand, et all., 2008, pp.28)

R and Spatial Objects

Every object in R is of a particular class. You have seen numeric, vector, matrix and data.frame classes. To learn the type of the object type in R:

```
> x=c(1,2,3,5)
> class(x)
[1] "numeric"
```

In R, for spatial data, the foundation class is the Spatial class, with just two slots. The first is a bounding box, a matrix of numerical coordinates with column names c(min,max), and at least two rows, with the first row eastings (x-axis) and the second northings (y-axis). The second is a CRS class object defining the coordinate reference system. (Bivand, et all., 2008, pp.28)
Start R and load the packages spdep and splancs.

R and Spatial Objects

Every object in R is of a particular class. You have seen numeric, vector, matrix and data.frame classes. To learn the type of the object type in R:

```
> x=c(1,2,3,5)
> class(x)
```

```
[1] "numeric"
```

In R, for spatial data, the foundation class is the Spatial class, with just two slots. The first is a bounding box, a matrix of numerical coordinates with column names c(min,max), and at least two rows, with the first row eastings (x-axis) and the second northings (y-axis). The second is a CRS class object defining the coordinate reference system. (Bivand, et all., 2008, pp.28)
Start R and load the packages spdep and splancs.

To see the available class of spatial objects type:

```
> getClass("Spatial")
```

```
Class "Spatial" [package "sp"]

Slots:

Name:         bbox proj4string
Class:      matrix         CRS

Known Subclasses:
Class "SpatialPoints", directly
Class "SpatialGrid", directly
Class "SpatialLines", directly
Class "SpatialPolygons", directly
Class "SpatialPointsDataFrame", by class "SpatialPoints", distance 2
Class "SpatialPixels", by class "SpatialPoints", distance 2
Class "SpatialGridDataFrame", by class "SpatialGrid", distance 2
Class "SpatialLinesDataFrame", by class "SpatialLines", distance 2
```

R and Spatial Objects

Most of the data used in economic research are irregular aeral data and generally consist of regional aggregates, which are administratively and politically defined. These can be cities, provinces, states, regions etc. Data belonging to such regional form are stored in a shapefile, which are produced by GIS.

R and Spatial Objects

Most of the data used in economic research are irregular aeral data and generally consist of regional aggregates, which are administratively and politically defined. These can be cities, provinces, states, regions etc. Data belonging to such regional form are stored in a shapefile, which are produced by GIS.

A shapefile consists of at least 3 files with extensions of .shp, .dbf, .shx. Using the readShapePoly function in R, it is easy to import, store and visualise a shapefile. Using the object created after importing the shapefile into R, spatial analysis can be used via the package spdep.

Example Data Sets that will be Used

Throughout the tutorials we will be using the famous Boston house price data,Columbus crime data, EU regional data collected at the NUTS2 level.

- Boston data: contains the classic Harrison and Rubinfeld (1978) housing data set with observations on 23 variables for 506 census tracts.

Example Data Sets that will be Used

Throughout the tutorials we will be using the famous Boston house price data,Columbus crime data, EU regional data collected at the NUTS2 level.

- Boston data: contains the classic Harrison and Rubinfeld (1978) housing data set with observations on 23 variables for 506 census tracts.
- Columbus data: http://www.neighborhoodscout.com/oh/columbus/crime/

Example Data Sets that will be Used

Throughout the tutorials we will be using the famous Boston house price data,Columbus crime data, EU regional data collected at the NUTS2 level.

- Boston data: contains the classic Harrison and Rubinfeld (1978) housing data set with observations on 23 variables for 506 census tracts.
- Columbus data: http://www.neighborhoodscout.com/oh/columbus/crime/
- EU2 Data: There are three levels of Nomenclature of Territorial Units for Statistics (NUTS) defined. This category refers to regions belonging to the second level (NUTS 2, also known as NUTS II), which is largely used by Eurostat and other European Union bodies.

Example Data Sets that will be Used

Throughout the tutorials we will be using the famous Boston house price data,Columbus crime data, EU regional data collected at the NUTS2 level.

- Boston data: contains the classic Harrison and Rubinfeld (1978) housing data set with observations on 23 variables for 506 census tracts.
- Columbus data: http://www.neighborhoodscout.com/oh/columbus/crime/
- EU2 Data: There are three levels of Nomenclature of Territorial Units for Statistics (NUTS) defined. This category refers to regions belonging to the second level (NUTS 2, also known as NUTS II), which is largely used by Eurostat and other European Union bodies.

Any spatial data set includes the geometric characteristics of the spatial units which are spatial coordinates, index of the geometries, the attributes associated to each spatial unit. These can be any economic, social data.

Example Data Sets that will be Used

Throughout the tutorials we will be using the famous Boston house price data,Columbus crime data, EU regional data collected at the NUTS2 level.

- Boston data: contains the classic Harrison and Rubinfeld (1978) housing data set with observations on 23 variables for 506 census tracts.
- Columbus data: http://www.neighborhoodscout.com/oh/columbus/crime/
- EU2 Data: There are three levels of Nomenclature of Territorial Units for Statistics (NUTS) defined. This category refers to regions belonging to the second level (NUTS 2, also known as NUTS II), which is largely used by Eurostat and other European Union bodies.

Any spatial data set includes the geometric characteristics of the spatial units which are spatial coordinates, index of the geometries, the attributes associated to each spatial unit. These can be any economic, social data.
The datasets are available from:

`http://www.isletme.istanbul.edu.tr/ogrelem/sebnemer/spatial/`

Columbus Data Set I

Now let's go to the datasets available in spdep R package, specifically Columbus data. To do that type in R:

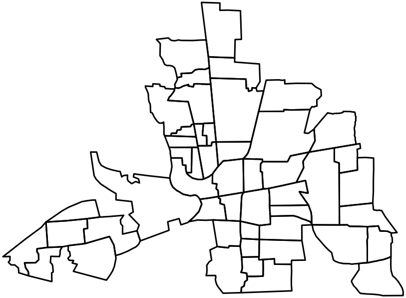
```
> example(columbus)

colmbs> columbus <- readShapePoly(system.file("etc/shapes/columbus.shp",
colmbs+  package="spdep")[1])

colmbs> col.gal.nb <- read.gal(system.file("etc/weights/columbus.gal",
colmbs+  package="spdep")[1])
```

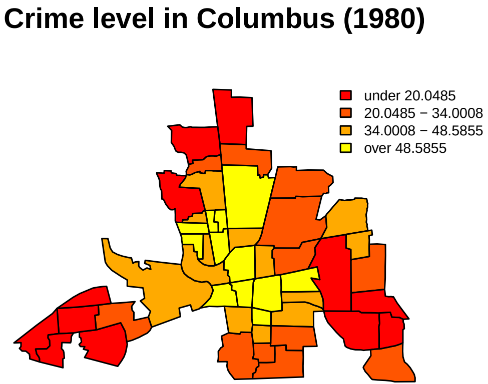
A major pleasure in working with spatial data is their visualisation. Maps are amongst the most compelling graphics. Let's plot the regions:

```
> plot(columbus)
```



To see the spread of CRIME among the regions:

```
> brks <- round(quantile(columbus$CRIME), digits=4)
> colours <- heat.colors(length(brks))
> plot(columbus, col=colours[findInterval(columbus$CRIME,
+ brks,all.inside=TRUE)])
> legend(x="topright", legend=leglabs(brks),fill=colours,
+ bty="n", cex=0.6)
> title(main="Crime level in Columbus (1980)",cex=.4)
```



How to Import Spatial Data

To import the shapefile for EU at the NUTS2 level, we will use the readShapePoly function as follows:

How to Import Spatial Data

To import the shapefile for EU at the NUTS2 level, we will use the readShapePoly function as follows: Assuming that the EU2 spatial data files are in your working directory:

```
> EU2=readShapePoly("EU2",IDvar="Id")
> getinfo.shape("EU2.shp")
```

```
Shapefile type: Polygon, (5), # of Shapes: 190
```

```
> names(EU2)
```

```
 [1] "SP_ID"      "Id"         "Name"       "gprb"
 [5] "lninv1b"    "pr80b"      "pr103b"     "lndens_emp"
 [9] "lndens_pop" "lnagrib"
```

How to Import Spatial Data

To import the shapefile for EU at the NUTS2 level, we will use the readShapePoly function as follows: Assuming that the EU2 spatial data files are in your working directory:

```
> EU2=readShapePoly("EU2",IDvar="Id")
> getinfo.shape("EU2.shp")
```

```
Shapefile type: Polygon, (5), # of Shapes: 190
```

```
> names(EU2)
```

```
 [1] "SP_ID"      "Id"         "Name"       "gprb"
 [5] "lninv1b"    "pr80b"      "pr103b"     "lndens_emp"
 [9] "lndens_pop" "lnagrib"
```

You may want to update this data with other variables included in the text file. Firstly, read in the text file into R.

How to Import Spatial Data

To import the shapefile for EU at the NUTS2 level, we will use the readShapePoly function as follows: Assuming that the EU2 spatial data files are in your working directory:

```
> EU2=readShapePoly("EU2",IDvar="Id")
> getinfo.shape("EU2.shp")
```

```
Shapefile type: Polygon, (5), # of Shapes: 190
```

```
> names(EU2)
```

```
 [1] "SP_ID"      "Id"          "Name"        "gprb"
 [5] "lninv1b"    "pr80b"       "pr103b"      "lndens_emp"
 [9] "lndens_pop" "lnagrib"
```

You may want to update this data with other variables included in the text file. Firstly, read in the text file into R.

```
> EU_updated=read.table("EU_updated.txt",header=T)
> names(EU_updated)
```

```
 [1] "Codes"   "Pop1990" "Pop1991" "Pop1992" "Pop1993" "Pop1994"
 [7] "Pop1995" "Pop1996" "Pop1997" "Pop1998" "Pop1999" "Pop2000"
[13] "Pop2001" "Pop2002" "Pop2003" "Pop2004" "Pop2005" "Pop2006"
[19] "Pop2007" "Pop2008" "Pop2009" "Pop2010" "Pop2011" "GDP1995"
[25] "GDP1996" "GDP1997" "GDP1998" "GDP1999" "GDP2000" "GDP2001"
[31] "GDP2002" "GDP2003" "GDP2004" "GDP2005" "GDP2006" "GDP2007"
[37] "GDP2008" "GDP2009"
```

How to Import Spatial Data

Then, merge the shapefile and the dataframe using the merge function according to the columns of Id in the EU2 shapefile and the Codes in the newEU dataframe.

```
> newEU=merge(EU2, EU_updated,by.x="Id",by.y="Codes")
> names(newEU)
```

```
 [1] "Id"         "SP_ID"      "Name"       "gprb"
 [5] "lninv1b"    "pr80b"      "pr103b"     "lndens_emp"
 [9] "lndens_pop" "lnagrib"    "Pop1990"    "Pop1991"
[13] "Pop1992"    "Pop1993"    "Pop1994"    "Pop1995"
[17] "Pop1996"    "Pop1997"    "Pop1998"    "Pop1999"
[21] "Pop2000"    "Pop2001"    "Pop2002"    "Pop2003"
[25] "Pop2004"    "Pop2005"    "Pop2006"    "Pop2007"
[29] "Pop2008"    "Pop2009"    "Pop2010"    "Pop2011"
[33] "GDP1995"    "GDP1996"    "GDP1997"    "GDP1998"
[37] "GDP1999"    "GDP2000"    "GDP2001"    "GDP2002"
[41] "GDP2003"    "GDP2004"    "GDP2005"    "GDP2006"
[45] "GDP2007"    "GDP2008"    "GDP2009"
```

How to Visualise Spatial Data

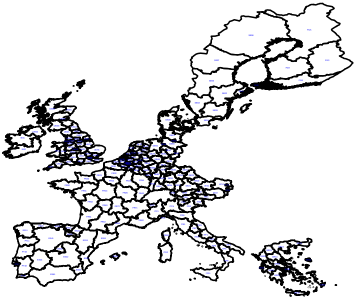
Let us plot the EU2 shapefile.

```
> plot(EU2)
```

How to Visualise Spatial Data

If you want to see the names of the NUTS2 levels then use the following:

```
> plot(EU2)
> text(coordinates(EU2),labels=sapply(slot(EU2,"polygons"),
+ function(i) slot(i,"ID")),cex=0.1,col="blue")
```



How to Visualise Spatial Data

Now let's calculate the quartiles for the GDP per capita in 2009.

```
> brks <- round(quantile(newEU$GDP2009), digits=4)
```

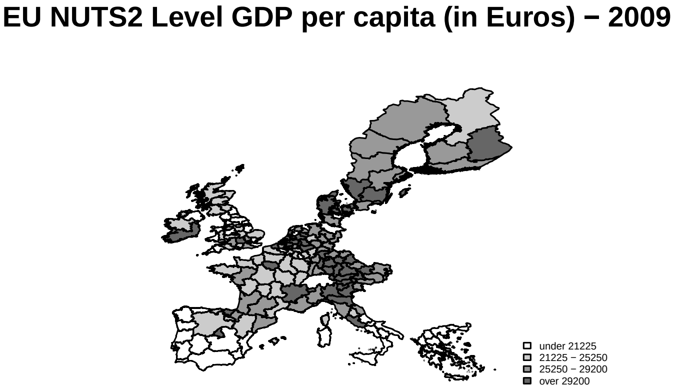
To give a grey scaled colour to each quartile, use the following:

```
> colours <- grey((length(brks):2)/length(brks))
```

How to Visualise Spatial Data

To plot these colours on the map, type the following:

```
> plot(EU2, col=colours[findInterval(newEU$GDP2009, brks,all.inside=TRUE)])
> legend(x="bottomright", legend=leglabs(brks),fill=colours, bty="n", cex=0.4)
> title(main="EU NUTS2 Level GDP per capita (in Euros) - 2009", cex=0.3)
```



Tobler's first law of geography:
"Everything is related to everything else, but close things are more related than things that are far apart."

Tobler's first law of geography:
"Everything is related to everything else, but close things are more related than things that are far apart."
From this definition, we need to identify who are the neighbours of each unit and what is the strength of their relationship. In order to do that for each unit we need a list of neighbours and spatial weights.

Creating Neighbours and Connectivity (Weights') Matrix

In order to analyse a spatial dataset, firstly, we need to do two things:

1. Defining spatial neighbours
2. Creating spatial weights' matrices

Defining Spatial Neighbours

Neighbourhood relationship can be defined in different ways: (Arbia, 2005, pp.37)

1. Contiguity-based neighbourhood:
   In the case of discrete-parameter random fields, a simple definition of neighbourhood could be based on the mere adjacency between 2 polygons. In this case two polygons indexed by S_i and S_j are said to be neighbours if they share a common boundary.
   This can be in the 3 different form:
   - Rook
   - Bishop
   - Queen

Defining Spatial Neighbours

Neighbourhood relationship can be defined in different ways: (Arbia, 2005, pp.37)

1. Contiguity-based neighbourhood:
   In the case of discrete-parameter random fields, a simple definition of neighbourhood could be based on the mere adjacency between 2 polygons. In this case two polygons indexed by S_i and S_j are said to be neighbours if they share a common boundary.
   This can be in the 3 different form:
   - Rook
   - Bishop
   - Queen
2. K-nearest neighbourhood
   Two sites S_i and S_j are said to be neighbours if $d_{ij} = Min(d_{ik}) \forall i, k$.

Defining Spatial Neighbours

Neighbourhood relationship can be defined in different ways: (Arbia, 2005, pp.37)

1. Contiguity-based neighbourhood:
   In the case of discrete-parameter random fields, a simple definition of neighbourhood could be based on the mere adjacency between 2 polygons. In this case two polygons indexed by S_i and S_j are said to be neighbours if they share a common boundary.
   This can be in the 3 different form:
   - Rook
   - Bishop
   - Queen
2. K-nearest neighbourhood
   Two sites S_i and S_j are said to be neighbours if $d_{ij} = Min(d_{ik}) \forall i, k$.
3. Critical cut-off neighbourhood
   Two sites S_i and S_j are said to be neighbours if $0 <= d_{ij} < d^*$, with d_{ij} the appropriate distance adopted, and d^* representing a critical cut-off.

Neighbours in R - Contiguity-based neighbourhood:

Neighbours are stored as an "nb" class. Very easy to remember nb:neighbours. In R:

Neighbours in R - Contiguity-based neighbourhood:

Neighbours are stored as an "nb" class. Very easy to remember nb:neighbours. In R:

```
> contnb_queen=poly2nb(columbus,queen=TRUE)
> contnb_queen[1]
```

```
[[1]]
[1] 2 3
```

```
> summary(contnb_queen)
```

```
Neighbour list object:
Number of regions: 49
Number of nonzero links: 236
Percentage nonzero weights: 9.829238
Average number of links: 4.816327
Link number distribution:

 2  3  4  5  6  7  8  9 10
 5  9 12  5  9  3  4  1  1
5 least connected regions:
0 5 41 45 46 with 2 links
1 most connected region:
19 with 10 links
```

```
> coords=coordinates(columbus) # to be able to see the centeroid
> # of each polygon
```

Neighbours in R

Let's say we want to see on the plot who is the neighbour of who:

```
> plot(columbus) # plots a simple map
> plot(contnb_queen,coords,add=TRUE) # adds lines of connectivity according to the co
```

Neighbours in R - K-nearest neighbourhood

There are 2 steps to be followed:

1. Firstly, we need to define the k nearest neighbours:
2. Secondly, we need to create the nb class of neighbours using the knn2nb().

```
> whoisthefirstnear=knearneigh(coords,k=1,longlat=TRUE)
> knn1columbus=knn2nb(whoisthefirstnear)
```

Neighbours in R - K-nearest neighbourhood

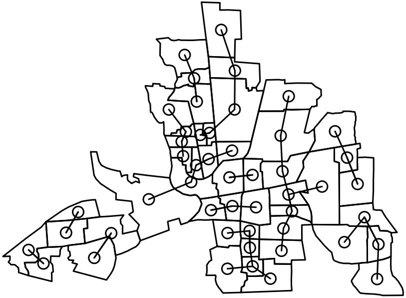
There are 2 steps to be followed:

1. Firstly, we need to define the k nearest neighbours:
2. Secondly, we need to create the nb class of neighbours using the knn2nb().

```
> whoisthefirstnear=knearneigh(coords,k=1,longlat=TRUE)
> knn1columbus=knn2nb(whoisthefirstnear)
```

If you want to plot the new neighbours on the plot simply repeat what you have done previously.

```
> plot(columbus) # plots a simple map
> plot(knn1columbus,coords,add=TRUE) # adds lines of connectivity according to the 1st nearest nb
```



Neighbours in R - Critical cut-off neighbourhood

In order to define the neighbours within a distance we need that distance measure. It is called a threshold.

Neighbours in R - Critical cut-off neighbourhood

In order to define the neighbours within a distance we need that distance measure. It is called a threshold. To calculate the threshold, we will use a function called nbdists() to calculate the distances between the centroids of the polygons. Let's say we base our calculations according to the 1st nearest neighbours.

Neighbours in R - Critical cut-off neighbourhood

In order to define the neighbours within a distance we need that distance measure. It is called a threshold. To calculate the threshold, we will use a function called nbdists() to calculate the distances between the centroids of the polygons. Let's say we base our calculations according to the 1st nearest neighbours.

```
> distBetwNeigh1=nbdists(knn1columbus,coords,longlat=TRUE)
> all.linkedTresh=max(unlist(distBetwNeigh1))
> all.linkedTresh
```

```
[1] 67.48852
```

Neighbours in R - Critical cut-off neighbourhood

In order to define the neighbours within a distance we need that distance measure. It is called a threshold. To calculate the threshold, we will use a function called nbdists() to calculate the distances between the centroids of the polygons. Let's say we base our calculations according to the 1st nearest neighbours.

```
> distBetwNeigh1=nbdists(knn1columbus,coords,longlat=TRUE)
> all.linkedTresh=max(unlist(distBetwNeigh1))
> all.linkedTresh
```

```
[1] 67.48852
```

```
> dnbTresh1=dnearneigh(coords,0,68,longlat=TRUE)
> summary(dnbTresh1)
```

```
Neighbour list object:
Number of regions: 49
Number of nonzero links: 252
Percentage nonzero weights: 10.49563
Average number of links: 5.142857
Link number distribution:

 1  2  3  4  5  6  7  8  9 10 11
 4  8  6  2  5  8  6  2  6  1  1
4 least connected regions:
6 10 21 47 with 1 link
1 most connected region:
28 with 11 links
```

Neighbours in R

If you want to plot the new neighbours on the plot simply repeat what you have done previously.

```
> plot(columbus) # plots a simple map
> plot(dnbTresh1,coords,add=TRUE) # adds lines of connectivity according to the 1st r
```

Neighbours in R

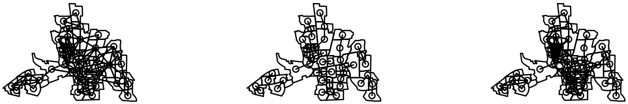
If you want to plot the new neighbours on the plot simply repeat what you have done previously.

```
> plot(columbus) # plots a simple map
> plot(dnbTresh1,coords,add=TRUE) # adds lines of connectivity according to the 1st r
```

Neighbours in R

You can plot the 3 neighbourhood definitions side-by-side. Remember par()?

```
> par (mfrow = c(1,3))
> plot(columbus)
> plot(contnb_queen,coords,add=TRUE)
> plot(columbus)
> plot(knn1columbus,coords,add=TRUE)
> plot(columbus)
> plot(dnbTresh1,coords,add=TRUE)
```



Defining Weights

We use nb2listw() to create the weights.
Let's say we are gonna create the weights' matrix according to the first type of neighbourhood relationship, contiguity based.

Defining Weights

We use nb2listw() to create the weights.
Let's say we are gonna create the weights' matrix according to the first type of neighbourhood relationship, contiguity based.

```
> contnb_queen.listw=nb2listw(contnb_queen,style="W",zero.policy=FALSE)
> class(contnb_queen.listw)

[1] "listw" "nb"

> contnb_queen[1] # polygon indices of the 1st one.

[[1]]
[1] 2 3

> contnb_queen.listw$weights[1]

[[1]]
[1] 0.5 0.5
```

Defining Weights

We use nb2listw() to create the weights.
Let's say we are gonna create the weights' matrix according to the first type of neighbourhood relationship, contiguity based.

```
> contnb_queen.listw=nb2listw(contnb_queen,style="W",zero.policy=FALSE)
> class(contnb_queen.listw)

[1] "listw" "nb"

> contnb_queen[1] # polygon indices of the 1st one.

[[1]]
[1] 2 3

> contnb_queen.listw$weights[1]

[[1]]
[1] 0.5 0.5
```

Now you can try to understand the others.

Morans I

Let's continue with the contiguity based neighbourhood definition:

```
> moran.test(columbus$CRIME,contnb_queen.listw)

Morans I test under randomisation

data:  columbus$CRIME
weights: contnb_queen.listw

Moran I statistic standard deviate = 5.5894, p-value =
1.139e-08
alternative hypothesis: greater
sample estimates:
Moran I statistic       Expectation          Variance
      0.500188557      -0.020833333       0.008689289
```

Please calculate the Morans I according to the other neighbourhood relationships.

Having your Own Spatial Weights Matrix in a File

In most cases, we have a data.frame and a weights matrix that is prepared by using other kinds of neighbourhood relationships rather than the ones based on distances.

Having your Own Spatial Weights Matrix in a File

In most cases, we have a data.frame and a weights matrix that is prepared by using other kinds of neighbourhood relationships rather than the ones based on distances.
In that case, you have 2 files:

1. Your data.frame
2. Your weights matrix

Having your Own Spatial Weights Matrix in a File

In most cases, we have a data.frame and a weights matrix that is prepared by using other kinds of neighbourhood relationships rather than the ones based on distances.
In that case, you have 2 files:

1. Your data.frame
2. Your weights matrix

Firstly, you need to upload them into R and then do a few manipulations in the weights' matrix. Then, you can do any kind of analysis.
Now let's see how to do this:

Having your Own Spatial Weights Matrix in a File

In most cases, we have a data.frame and a weights matrix that is prepared by using other kinds of neighbourhood relationships rather than the ones based on distances.
In that case, you have 2 files:

1. Your data.frame
2. Your weights matrix

Firstly, you need to upload them into R and then do a few manipulations in the weights' matrix. Then, you can do any kind of analysis.
Now let's see how to do this:

```
> gdpTR12region=read.table("turkey12regionsGDP2001.txt", header=TRUE)
> TR12regionWM=as.matrix(read.table("turkey12regionsWeightsMatrix.txt"))
> weights.listw=mat2listw(TR12regionWM, style="W")
> weights.listw$weights[1] #for Istanbul

[[1]]
[1] 0.2 0.2 0.2 0.2 0.2

> moran.test(gdpTR12region$GDPindollars2001,weights.listw)

Morans I test under randomisation


data:  gdpTR12region$GDPindollars2001
weights: weights.listw


Moran I statistic standard deviate = 3.9918, p-value =
3.279e-05
alternative hypothesis: greater
sample estimates:
Moran I statistic       Expectation          Variance
       0.51524743       -0.09090909        0.02305872
```

OLS vs Spatial Analysis

If you do OLS to a dataset that has spatial dependence, then you are in trouble. To see if your OLS residuals have a spatial dependence you can simply run an OLS and obtain the residuals and test your residuals with Morans I test.

```
> OLScolumbus=lm(CRIME~INC+HOVAL, data=columbus)
> plot(residuals(OLScolumbus))
> moran.test(residuals(OLScolumbus),contnb_queen.listw)
```

```
Morans I test under randomisation

data:  residuals(OLScolumbus)
weights: contnb_queen.listw

Moran I statistic standard deviate = 2.6521, p-value =
0.003999
alternative hypothesis: greater
sample estimates:
Moran I statistic       Expectation          Variance
      0.222109407      -0.020833333       0.008391173
```

OLS vs Spatial Analysis

To apply the Morans I test in the OLS residuals use the function lm.morantest().

```
> OLScolumbus=lm(CRIME~INC+HOVAL, data=columbus)
> lm.morantest(OLScolumbus,contnb_queen.listw,resfun=rstudent)
```

```
Global Morans I for regression residuals

data:
model: lm(formula = CRIME ~ INC + HOVAL, data = columbus)
weights: contnb_queen.listw

Moran I statistic standard deviate = 2.355, p-value =
0.009262
alternative hypothesis: greater
sample estimates:
Observed Morans I        Expectation          Variance
      0.178520703       -0.033418335       0.008099305
```

OLS vs Spatial Analysis

To apply the Morans I test in the OLS residuals use the function lm.morantest().

```
> OLScolumbus=lm(CRIME~INC+HOVAL, data=columbus)
> lm.morantest(OLScolumbus,contnb_queen.listw,resfun=rstudent)

Global Morans I for regression residuals

data:
model: lm(formula = CRIME ~ INC + HOVAL, data = columbus)
weights: contnb_queen.listw

Moran I statistic standard deviate = 2.355, p-value =
0.009262
alternative hypothesis: greater
sample estimates:
Observed Morans I       Expectation          Variance
      0.178520703      -0.033418335       0.008099305
```

Since Morans I test is based on the assumption of normality, we'd better check if the residuals have a normal distribution or not.

Non-Spatial Tests for OLS Residuals

In order to run the non-spatial tests for OLS residuals, you need 2 packages:

```
> library(lmtest)
> library(tseries)
```

Heteroscedasticity Tests

```
> BP<-bptest(OLScolumbus, studentize=FALSE)
> BP

Breusch-Pagan test

data:  OLScolumbus
BP = 10.0128, df = 2, p-value = 0.006695

> BP<-bptest(OLScolumbus, studentize=TRUE)
> BP

studentized Breusch-Pagan test

data:  OLScolumbus
BP = 7.2166, df = 2, p-value = 0.0271
```

Results show that OLS resids are heteroscedastic

Normality Tests and Plots

Jarque-Bera Normality Test, Not Jarque-Beta!!!

Normality Tests and Plots

Jarque-Bera Normality Test, Not Jarque-Beta!!!

```
> JB<-jarque.bera.test(residuals(OLScolumbus))
> JB
```

```
Jarque Bera Test

data:  residuals(OLScolumbus)
X-squared = 1.8358, df = 2, p-value = 0.3994
```

Normality Tests and Plots

Jarque-Bera Normality Test, Not Jarque-Beta!!!

```
> JB<-jarque.bera.test(residuals(OLScolumbus))
> JB
```

```
Jarque Bera Test

data:  residuals(OLScolumbus)
X-squared = 1.8358, df = 2, p-value = 0.3994
```

To plot the residuals:

Normality Tests and Plots

Jarque-Bera Normality Test, Not Jarque-Beta!!!

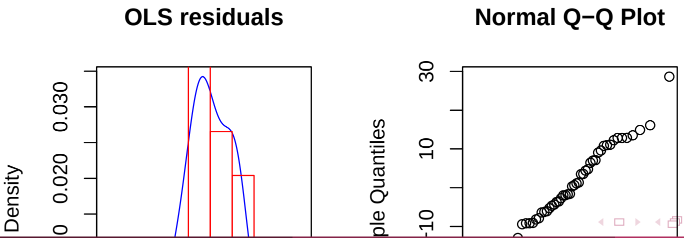
```
> JB<-jarque.bera.test(residuals(OLScolumbus))
> JB
```

```
Jarque Bera Test

data:  residuals(OLScolumbus)
X-squared = 1.8358, df = 2, p-value = 0.3994
```

To plot the residuals:

```
> par (mfrow = c(1,2))
> plot(density(residuals(OLScolumbus)), main="OLS residuals", col=4)
> hist(residuals(OLScolumbus), freq=FALSE, add=TRUE, border=2)
> qqnorm(residuals(OLScolumbus))
```



Spatial Tests

Morans I cannot distinguish between a SAR or S-Err model. Anselin (1988) suggests Lagrange Multiplier Tests that will help us find out which model should be estimated.

Spatial Tests

Morans I cannot distinguish between a SAR or S-Err model. Anselin (1988) suggests Lagrange Multiplier Tests that will help us find out which model should be estimated.

```
> ST<-lm.LMtests(OLScolumbus, listw = contnb_queen.listw, test = "all")
> out<-t(sapply(ST, function(x) c(x$statistic, x$parameter, x$p.value)))
> colnames(out)<- c("Statistics", "df", "p-value")
> printCoefmat(out)
```

```
       Statistics       df p-value
LMerr    5.206214 1.000000  0.0225
LMlag    8.897999 1.000000  0.0029
RLMerr   0.043906 1.000000  0.8340
RLMlag   3.735691 1.000000  0.0533
SARMA    8.941905 2.000000  0.0114
```

Spatial Lag Model - SAR

The spatial lag model is as follows:

$$Y = \rho WY + X\beta + \epsilon$$

Spatial Lag Model - SAR

The spatial lag model is as follows:

$$Y = \rho WY + X\beta + \epsilon$$

Since there is the lagged Y in the model, we cannot estimate this model with OLS. We can either use,

- Maximum Likelihood

Spatial Lag Model - SAR

The spatial lag model is as follows:

$$Y = \rho WY + X\beta + \epsilon$$

Since there is the lagged Y in the model, we cannot estimate this model with OLS. We can either use,

- Maximum Likelihood
  or
- Instrumental Variables

SAR - ML

The estimation of a SAR model with ML in R with different methods is done with the following code:

```
lagsarlm(model, data, weightsmatrix, method)
method={eigen, Matrix, spam, LU, Chebyshev, MC}
```

SAR - ML

The estimation of a SAR model with ML in R with different methods is done with the following code:

```
lagsarlm(model, data, weightsmatrix, method)
method={eigen, Matrix, spam, LU, Chebyshev, MC}
```

```
> sarml.eigColumbus<-lagsarlm(CRIME ~ INC + HOVAL,data = columbus,
+ listw = contnb_queen.listw, method = "eigen")
> summary(sarml.eigColumbus)
Call:
lagsarlm(formula = CRIME ~ INC + HOVAL, data = columbus, listw = contnb_queen.listw,
    method = "eigen")

Residuals:
       Min         1Q     Median         3Q        Max
-37.652017  -5.334611   0.071473   6.107196  23.302618

Type: lag
Coefficients: (asymptotic standard errors)
             Estimate Std. Error z value  Pr(>|z|)
(Intercept) 45.603250   7.257404  6.2837 3.306e-10
INC         -1.048728   0.307406 -3.4115  0.000646
HOVAL       -0.266335   0.089096 -2.9893  0.002796

Rho: 0.42333, LR test value: 9.4065, p-value: 0.0021621
Asymptotic standard error: 0.11951
    z-value: 3.5422, p-value: 0.00039686
Wald statistic: 12.547, p-value: 0.00039686

Log likelihood: -182.674 for lag model
ML residual variance (sigma squared): 96.857, (sigma: 9.8416)
```

SAR - Mixed Model

You may want to include the lagged explanatory variables in your model:

SAR - Mixed Model

You may want to include the lagged explanatory variables in your model:

```
> sarml.mixedColumbus<-lagsarlm(CRIME ~ INC + HOVAL,data = columbus,
+ listw = contnb_queen.listw, type = "mixed")
> summary(sarml.mixedColumbus)
```

```
Call:
lagsarlm(formula = CRIME ~ INC + HOVAL, data = columbus, listw = contnb_queen.listw,
    type = "mixed")

Residuals:
      Min        1Q    Median        3Q       Max
-37.31199  -6.49556  -0.22971   6.17872  22.74795

Type: mixed
Coefficients: (asymptotic standard errors)
             Estimate Std. Error z value  Pr(>|z|)
(Intercept) 44.320005  13.045474  3.3973 0.0006804
INC         -0.919906   0.334742 -2.7481 0.0059941
HOVAL       -0.297129   0.090416 -3.2863 0.0010153
lag.INC     -0.583913   0.574225 -1.0169 0.3092139
lag.HOVAL    0.257684   0.187235  1.3763 0.1687404

Rho: 0.40346, LR test value: 4.6627, p-value: 0.030825
Asymptotic standard error: 0.16133
    z-value: 2.5008, p-value: 0.012392
Wald statistic: 6.254, p-value: 0.012392

Log likelihood: -181.6393 for mixed model
ML residual variance (sigma squared): 93.272, (sigma: 9.6578)
```

SAR - Instrumental Variables

The estimation of a SAR model in R with the Instrumental Variables method is done with the following code:

```
stsls(model, data, weightsmatrix)
```

SAR - Instrumental Variables

The estimation of a SAR model in R with the Instrumental Variables method is done with the following code:

```
stsls(model, data, weightsmatrix)
```

```
> stslsColumbus<-stsls(CRIME ~ INC + HOVAL,data = columbus,
+ listw = contnb_queen.listw)
> summary(stslsColumbus)

Call:
stsls(formula = CRIME ~ INC + HOVAL, data = columbus, listw = contnb_queen.listw)

Residuals:
      Min        1Q    Median        3Q       Max
-37.94358 -5.64216 -0.24203   6.22748  22.82069

Coefficients:
             Estimate Std. Error t value  Pr(>|t|)
Rho          0.461487   0.187939  2.4555  0.014069
(Intercept) 43.528473  11.061569  3.9351 8.316e-05
INC         -0.999276   0.385591 -2.5915  0.009555
HOVAL       -0.265650   0.092391 -2.8753  0.004037

Residual variance (sigma squared): 104.66, (sigma: 10.231)
```

Spatial Error Model - SE

The spatial error model is as follows:

$$Y = X\beta + \lambda W\epsilon + u$$

The OLS will be inefficient so we have to use the following methods:

1. ML
2. GM

SE - ML

This is done in R with the following code:

```
errorsarlm(model, data, weightsmatrix, method)
method={eigen, Matrix, spam, LU, Chebyshev, MC}
```

SE - ML

This is done in R with the following code:

```
errorsarlm(model, data, weightsmatrix, method)
method={eigen, Matrix, spam, LU, Chebyshev, MC}
```

```
> errorsarml.eigColumbus<-errorsarlm(CRIME ~ INC + HOVAL,data = columbus,
+ listw = contnb_queen.listw, method = "eigen")
> summary(errorsarml.eigColumbus)
Call:
errorsarlm(formula = CRIME ~ INC + HOVAL, data = columbus, listw = contnb_queen.listw,
    method = "eigen")

Residuals:
      Min        1Q    Median        3Q       Max
-34.65998  -6.16943  -0.70623   7.75392  23.43878

Type: error
Coefficients: (asymptotic standard errors)
             Estimate Std. Error z value  Pr(>|z|)
(Intercept) 60.279469   5.365594 11.2344 < 2.2e-16
INC         -0.957305   0.334231 -2.8642 0.0041806
HOVAL       -0.304559   0.092047 -3.3087 0.0009372

Lambda: 0.54675, LR test value: 7.2556, p-value: 0.0070679
Asymptotic standard error: 0.13805
    z-value: 3.9605, p-value: 7.4786e-05
Wald statistic: 15.686, p-value: 7.4786e-05

Log likelihood: -183.7494 for error model
```

SE - GMM

This is done in R with the following code:

```
GMerrorsar(model, data, weightsmatrix)
```

SE - GMM

This is done in R with the following code:

```
GMerrorsar(model, data, weightsmatrix)
```

```
> GMerrorsar.Columbus<-GMerrorsar(CRIME ~ INC + HOVAL,data = columbus,
+ listw = contnb_queen.listw, method = "nlminb")
> summary(GMerrorsar.Columbus)

Call:
GMerrorsar(formula = CRIME ~ INC + HOVAL, data = columbus, listw = contnb_queen.listw,
    method = "nlminb")

Residuals:
     Min       1Q   Median       3Q      Max
-30.6689  -6.6664  -2.3305   9.8505  28.6764

Type: GM SAR estimator
Coefficients: (GM standard errors)
             Estimate Std. Error z value  Pr(>|z|)
(Intercept) 62.918810   5.111018 12.3104 < 2.2e-16
INC         -1.150075   0.341405 -3.3687 0.0007554
HOVAL       -0.298231   0.096707 -3.0839 0.0020434

Lambda: 0.38345 (standard error): 0.42729 (z-value): 0.89742
Residual variance (sigma squared): 108.18, (sigma: 10.401)
GM argmin sigma squared: 107.84
Number of observations: 49
Number of parameters estimated: 5
```

SARAR Model - ML

```
sacsarlm(model, data, weightsmatrix)
```

SARAR Model - ML

sacsarlm(model, data, weightsmatrix)

```
> sacsarlm.Columbus<-sacsarlm(CRIME ~ INC + HOVAL,data = columbus,
+ listw = contnb_queen.listw)
> summary(sacsarlm.Columbus)
Call:
sacsarlm(formula = CRIME ~ INC + HOVAL, data = columbus, listw = contnb_queen.listw)

Residuals:
      Min        1Q    Median        3Q       Max
-37.35601  -5.04457  -0.18999   6.71177  23.28400

Type: sac
Coefficients: (asymptotic standard errors)
             Estimate Std. Error z value  Pr(>|z|)
(Intercept) 47.915359   9.985548  4.7985 1.599e-06
INC         -1.042749   0.328628 -3.1730  0.001509
HOVAL       -0.279841   0.090699 -3.0854  0.002033

Rho: 0.36937
Asymptotic standard error: 0.19625
    z-value: 1.8821, p-value: 0.059817
Lambda: 0.14642
Asymptotic standard error: 0.30102
    z-value: 0.4864, p-value: 0.62668

LR test value: 9.6444, p-value: 0.0080489
```

SARAR Model - GMM

```
gstsls(model, data, weightsmatrix)
```

SARAR Model - GMM

```
gstsls(model, data, weightsmatrix)
```

```
> gstsls.Columbus<-gstsls(CRIME ~ INC + HOVAL,data = columbus,
+ listw = contnb_queen.listw)
> summary(gstsls.Columbus)

Call:
gstsls(formula = CRIME ~ INC + HOVAL, data = columbus, listw = contnb_queen.listw)

Residuals:
      Min        1Q    Median        3Q       Max
-37.92539  -5.54440  -0.15737   6.11962  22.88118

Type: GM SARAR estimator
Coefficients: (GM standard errors)
             Estimate Std. Error z value  Pr(>|z|)
Rho_Wy       0.461787   0.187391  2.4643  0.013728
(Intercept) 43.540444  11.090741  3.9258 8.643e-05
INC         -1.005003   0.386718 -2.5988  0.009355
HOVAL       -0.264092   0.092227 -2.8635  0.004190

Lambda: -0.016981
Residual variance (sigma squared): 104.72, (sigma: 10.233)
GM argmin sigma squared: 95
Number of observations: 49
Number of parameters estimated: 6
```

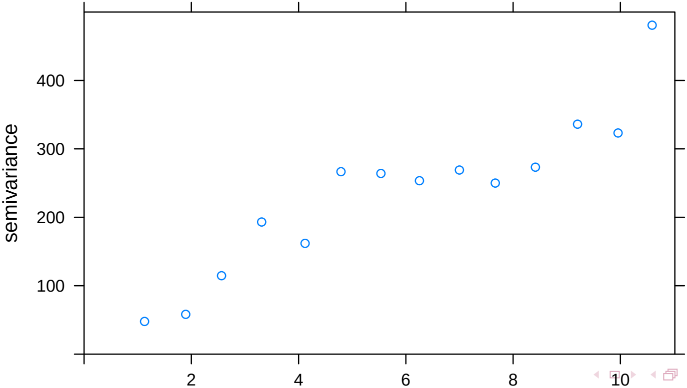
Estimating Spatial Correlation: Variogram

In geostatistics the spatial correlation is modelled by the variogram instead of a correlogram or covariogram. The variogram plots semivariance as a function of distance. (Bivand, et all., pp.195)
A simple way to acknowledge that spatial correlation is present or not is to make scatter plots of pairs $Z(s_i)$ and $Z(s_j)$, grouped according to their separation distance $h_{ij} = ||s_i - s_j||$.
The empirical semivariance is described by

$$\hat{\gamma}(h) = \frac{1}{2}\frac{1}{n_h}\sum_{i=1}^{n(h)}(z(s_i + h) - z(s_i))^2$$

Plotting Variogram

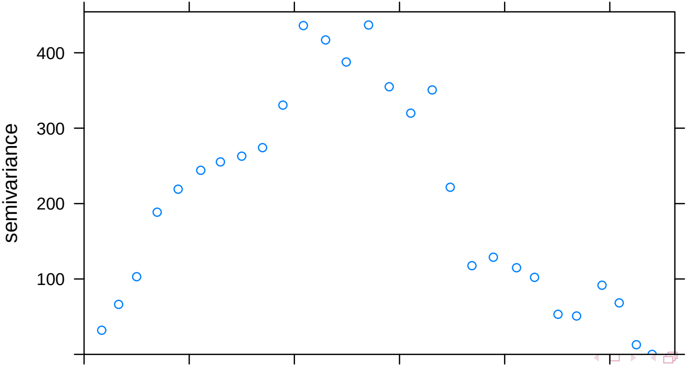
```
> library(gstat)
> data(columbus)
> # convert simple data frame into a spatial data frame object:
> coordinates(columbus) <- ~ X+Y
> variog1 <- variogram(CRIME~1, columbus)
> plot(variog1) # See 14 points
```



Plotting Variogram

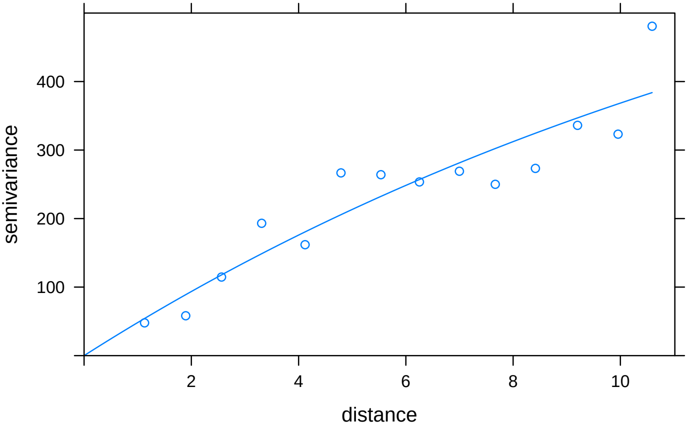
For the interval width, gstat uses a default of the cutoff value divided by 15.
Choosing a smaller interval width will result in more detail, as more estimates of (h) appear.
The distance vector does not have to be cut in regular intervals,
boundaries=c(0,50,100,seq(250,1500,250))

```
> variog2 <- variogram(CRIME~1, boundaries=0:100, data=columbus)
> # boundaries;number of points
> plot(variog2)
```



Plotting Variogram

```
> model.variog <- vgm(psill=200, model="Exp",range=5)
> fit.variog <- fit.variogram(variog1, model.variog)
> plot(variog1, model=fit.variog)
```



Plotting Variogram

The following parameters are often used to describe variograms:

- nugget : The height of the jump of the semivariogram at the discontinuity at the origin.
- sill : Limit of the variogram tending to infinity lag distances.
- range : The distance in which the difference of the variogram from the sill becomes negligible. In models with a fixed sill, it is the distance at which this is first reached; for models with an asymptotic sill, it is conventionally taken to be the distance when the semivariance first reaches 95 % of the sill..

psill=350-50 50
Model options are: exponential, spherical, Gaussian and Matern For weighted least squares fitting a variogram model to the sample variogram (Cressie, 1985), we need to take several steps:

1. Choose a suitable model (such as exponential, ...), with or without nugget
2. Choose suitable initial values for partial sill(s), range(s), and possibly nugget
3. Fit this model, using one of the fitting criteria.

Fit ranges and/or sills from a simple or nested variogram model to a sample variogram

Boston Data Set and the Variables

The original data are 506 observations on 14 variables, medv being the target variable.

crim	per capita crime rate by town
zn	proportion of residential land zoned for lots over 25,000 sq.ft
indus	proportion of non-retail business acres per town
chas	Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
nox	nitric oxides concentration (parts per 10 million)
rm	average number of rooms per dwelling
age	proportion of owner-occupied units built prior to 1940
dis	weighted distances to five Boston employment centres
rad	index of accessibility to radial highways
tax	full-value property-tax rate per USD 10,000
ptratio	pupil-teacher ratio by town
b	1000(B - 0.63)$^2 whereBistheproportionofblacksbytown$
lstat	percentage of lower status of the population
medv	median value of owner-occupied homes in USD 1000's

Boston Data Set and the Variables

The original data are 506 observations on 14 variables, medv being the target variable.

crim	per capita crime rate by town
zn	proportion of residential land zoned for lots over 25,000 sq.ft
indus	proportion of non-retail business acres per town
chas	Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
nox	nitric oxides concentration (parts per 10 million)
rm	average number of rooms per dwelling
age	proportion of owner-occupied units built prior to 1940
dis	weighted distances to five Boston employment centres
rad	index of accessibility to radial highways
tax	full-value property-tax rate per USD 10,000
ptratio	pupil-teacher ratio by town
b	1000(B - 0.63)$^2 whereBistheproportionofblacksbytown$
lstat	percentage of lower status of the population
medv	median value of owner-occupied homes in USD 1000's

The corrected data set has the following additional columns:

Boston Data Set and the Variables

The original data are 506 observations on 14 variables, medv being the target variable.

crim	per capita crime rate by town
zn	proportion of residential land zoned for lots over 25,000 sq.ft
indus	proportion of non-retail business acres per town
chas	Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
nox	nitric oxides concentration (parts per 10 million)
rm	average number of rooms per dwelling
age	proportion of owner-occupied units built prior to 1940
dis	weighted distances to five Boston employment centres
rad	index of accessibility to radial highways
tax	full-value property-tax rate per USD 10,000
ptratio	pupil-teacher ratio by town
b	1000(B - 0.63)$^2 whereBistheproportionofblacksbytown$
lstat	percentage of lower status of the population
medv	median value of owner-occupied homes in USD 1000's

The corrected data set has the following additional columns:

cmedv	corrected median value of owner-occupied homes in USD 1000's
town	name of town
tract	census tract
lon	longitude of census tract
lat	latitude of census tract

Spatial Weights Matrix and Morans I

Load the data set Boston:

```
> data(boston)
```

Spatial Weights Matrix and Morans I

Load the data set Boston:

```
> data(boston)
```

If you do ls() you can see what has been extracted from the data(boston) command.

Spatial Weights Matrix and Morans I

Load the data set Boston:

```
> data(boston)
```

If you do ls() you can see what has been extracted from the data(boston) command.
Create the weights matrix using:

```
> Boston.listw<-nb2listw(boston.soi)
```

Spatial Weights Matrix and Morans I

Load the data set Boston:

```
> data(boston)
```

If you do ls() you can see what has been extracted from the data(boston) command. Create the weights matrix using:

```
> Boston.listw<-nb2listw(boston.soi)
```

Estimate the Morans I for the CMEDV variable:

Spatial Weights Matrix and Morans I

Load the data set Boston:

> data(boston)

If you do ls() you can see what has been extracted from the data(boston) command.
Create the weights matrix using:

> Boston.listw<-nb2listw(boston.soi)

Estimate the Morans I for the CMEDV variable:

> moran.test(boston.c$CMEDV,Boston.listw)

```
Morans I test under randomisation

data:  boston.c$CMEDV
weights: Boston.listw

Moran I statistic standard deviate = 21.7861, p-value <
2.2e-16
alternative hypothesis: greater
sample estimates:
Moran I statistic       Expectation          Variance
      0.690285059      -0.001980198       0.001009685
```

Estimate an OLS for Boston Data and Test the Residuals

Now estimate an OLS for Boston Data and calculate the Morans I for the residuals. Use the following explanatory variables as it is:
$CRIM + CHAS + RM + AGE + log(RAD) + TAX + PTRATIO + log(LSTAT)$

Estimate an OLS for Boston Data and Test the Residuals

Now estimate an OLS for Boston Data and calculate the Morans I for the residuals. Use the following explanatory variables as it is:
$CRIM + CHAS + RM + AGE + log(RAD) + TAX + PTRATIO + log(LSTAT)$

```
> OLS.Boston<-lm(log(CMEDV) ~ CRIM + CHAS + RM +  AGE +
+ log(RAD) + TAX + PTRATIO + log(LSTAT), data = boston.c)
> summary(OLS.Boston)

Call:
lm(formula = log(CMEDV) ~ CRIM + CHAS + RM + AGE + log(RAD) +
    TAX + PTRATIO + log(LSTAT), data = boston.c)

Residuals:
     Min       1Q   Median       3Q      Max
-0.64623 -0.10960 -0.01081  0.10579  0.92289

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  4.0708184  0.1683797  24.176  < 2e-16 ***
CRIM        -0.0107837  0.0012566  -8.582  < 2e-16 ***
CHAS1        0.0917909  0.0346228   2.651 0.008277 **
RM           0.0649290  0.0172213   3.770 0.000183 ***
AGE          0.0012585  0.0004188   3.005 0.002789 **
log(RAD)     0.0792015  0.0191374   4.139 4.10e-05 ***
TAX         -0.0004889  0.0001059  -4.616 4.99e-06 ***
PTRATIO     -0.0235469  0.0046740  -5.038 6.60e-07 ***
log(LSTAT)  -0.4267376  0.0253077 -16.862  < 2e-16 ***
---
Signif. codes:  0 Ś***Š 0.001 Ś**Š 0.01 Ś*Š 0.05 Ś.Š 0.1 Ś Š 1

Residual standard error: 0.1924 on 497 degrees of freedom
Multiple R-squared: 0.7815, Adjusted R-squared: 0.778
F-statistic: 222.2 on 8 and 497 DF,  p-value: < 2.2e-16
> lm.morantest(OLS.Boston,Boston.listw)
Global Morans I for regression residuals

data:
model: lm(formula = log(CMEDV) ~ CRIM + CHAS + RM + AGE +
log(RAD) + TAX + PTRATIO + log(LSTAT), data = boston.c)
weights: Boston.listw
```

Estimate an OLS for Boston Data and Test the Residuals

Hint: You need the lmtest and tseries libraries!

Estimate an OLS for Boston Data and Test the Residuals

Hint: You need the lmtest and tseries libraries!
Heterosckedasticity: BP Test

Estimate an OLS for Boston Data and Test the Residuals

Hint: You need the lmtest and tseries libraries!
Heterosckedasticity: BP Test

```
> BP<-bptest(OLS.Boston, studentize=FALSE)
> BP

Breusch-Pagan test

data:  OLS.Boston
BP = 142.4877, df = 8, p-value < 2.2e-16
```

Estimate an OLS for Boston Data and Test the Residuals

Hint: You need the lmtest and tseries libraries!
Heterosckedasticity: BP Test

```
> BP<-bptest(OLS.Boston, studentize=FALSE)
> BP

Breusch-Pagan test

data:  OLS.Boston
BP = 142.4877, df = 8, p-value < 2.2e-16
```

Normality: JB Test

Estimate an OLS for Boston Data and Test the Residuals

Hint: You need the lmtest and tseries libraries!
Heterosckedasticity: BP Test

```
> BP<-bptest(OLS.Boston, studentize=FALSE)
> BP

Breusch-Pagan test

data:  OLS.Boston
BP = 142.4877, df = 8, p-value < 2.2e-16
```

Normality: JB Test

```
> JB<-jarque.bera.test(residuals(OLS.Boston))
> JB

Jarque Bera Test

data:  residuals(OLS.Boston)
X-squared = 151.034, df = 2, p-value < 2.2e-16
```

You can also examine the histogram and the QQ Plots.

Spatial Tests

Morans I for the OLS residuals indicate a spatial dependence but which way around is it? Is it a spatial lag, error or both?

Spatial Tests

Morans I for the OLS residuals indicate a spatial dependence but which way around is it? Is it a spatial lag, error or both?
Now do the spatial tests:

Spatial Tests

Morans I for the OLS residuals indicate a spatial dependence but which way around is it? Is it a spatial lag, error or both?
Now do the spatial tests:

```
> ST<-lm.LMtests(OLS.Boston, listw = Boston.listw ,test = "all")
> out<-t(sapply(ST, function(x) c(x$statistic, x$parameter, x$p.value)))
> colnames(out)<- c("Statistics", "df", "p-value")
> printCoefmat(out)

       Statistics      df p-value
LMerr     246.679   1.000       0
LMlag     216.405   1.000       0
RLMerr     61.158   1.000       0
RLMlag     30.884   1.000       0
SARMA     277.564   2.000       0
```

Spatial Lag Model - ML estimation

Remember the lagsarlm() function with different methods?

Spatial Lag Model - ML estimation

Remember the lagsarlm() function with different methods?

```
> sarml.eigBoston<-lagsarlm(log(CMEDV) ~ CRIM + CHAS + RM +  AGE +
+ log(RAD) + TAX + PTRATIO + log(LSTAT), data = boston.c,
+ listw = Boston.listw, method = "eigen")
> summary(sarml.eigBoston)
Call:
lagsarlm(formula = log(CMEDV) ~ CRIM + CHAS + RM + AGE + log(RAD) +
    TAX + PTRATIO + log(LSTAT), data = boston.c, listw = Boston.listw,
    method = "eigen")

Residuals:
       Min         1Q     Median         3Q        Max
-0.4739209 -0.0853567 -0.0096987  0.0854454  0.7837545

Type: lag
Coefficients: (asymptotic standard errors)
               Estimate  Std. Error  z value  Pr(>|z|)
(Intercept)  1.8537e+00  1.7859e-01  10.3795 < 2.2e-16
CRIM        -6.1513e-03  9.8682e-04  -6.2334 4.563e-10
CHAS1        1.4085e-02  2.6706e-02   0.5274 0.5979109
RM           6.9365e-02  1.3304e-02   5.2139 1.849e-07
AGE          1.1614e-03  3.2236e-04   3.6027 0.0003149
log(RAD)     6.1160e-02  1.4780e-02   4.1381 3.501e-05
TAX         -2.9047e-04  8.2982e-05  -3.5005 0.0004645
PTRATIO     -1.1570e-02  3.6739e-03  -3.1492 0.0016372
log(LSTAT)  -2.6294e-01  2.1163e-02 -12.4245 < 2.2e-16

Rho: 0.50301, LR test value: 221.1, p-value: < 2.22e-16
Asymptotic standard error: 0.030358
    z-value: 16.57, p-value: < 2.22e-16
Wald statistic: 274.55, p-value: < 2.22e-16

Log likelihood: 231.1388 for lag model
ML residual variance (sigma squared): 0.021826, (sigma: 0.14774)
Number of observations: 506
Number of parameters estimated: 11
AIC: -440.28, (AIC for lm: -221.17)
LM test for residual autocorrelation
```

Including the lagged values of the neighbours explanatory variables

If you think that the house prices in one region could be depending on the explanatory variables of the neighbours as well as the region's itself, estimate a mixed model.

Including the lagged values of the neighbours explanatory variables

If you think that the house prices in one region could be depending on the explanatory variables of the neighbours as well as the region's itself, estimate a mixed model.

```
> sarml.Bostonmix<-lagsarlm(log(CMEDV) ~CRIM + CHAS + RM +  AGE +
+ log(RAD) + TAX + PTRATIO + log(LSTAT),data=boston.c,
+ listw=Boston.listw, type="mixed")
> summary(sarml.Bostonmix)
```

```
Call:
lagsarlm(formula = log(CMEDV) ~ CRIM + CHAS + RM + AGE + log(RAD) +
    TAX + PTRATIO + log(LSTAT), data = boston.c, listw = Boston.listw,
    type = "mixed")

Residuals:
       Min         1Q     Median         3Q        Max
-0.6120096 -0.0727708 -0.0037606  0.0762290  0.7070618

Type: mixed
Coefficients: (asymptotic standard errors)
                  Estimate  Std. Error  z value  Pr(>|z|)
(Intercept)     1.24993827  0.20970235   5.9605 2.514e-09
CRIM           -0.00546618  0.00096093  -5.6884 1.282e-08
CHAS1          -0.03889090  0.02867550  -1.3562  0.175022
RM              0.08641534  0.01349227   6.4048 1.506e-10
AGE            -0.00076750  0.00050620  -1.5162  0.129469
log(RAD)        0.06131583  0.02308403   2.6562  0.007903
TAX            -0.00054042  0.00011544  -4.6812 2.851e-06
PTRATIO        -0.01483543  0.00612407  -2.4225  0.015415
log(LSTAT)     -0.28533252  0.02269912 -12.5702 < 2.2e-16
lag.CRIM       -0.00196619  0.00166123  -1.1836  0.236583
lag.CHAS1       0.10537268  0.04212312   2.5015  0.012365
lag.RM         -0.04704463  0.01923096  -2.4463  0.014433
lag.AGE         0.00162099  0.00060749   2.6683  0.007623
lag.log(RAD)   -0.02866263  0.03054088  -0.9385  0.347987
lag.TAX         0.00046616  0.00015473   3.0127  0.002589
lag.PTRATIO     0.00798781  0.00758700   1.0528  0.292420
lag.log(LSTAT)  0.12405259  0.03417903   3.6295  0.000284
```

Spatial Lag Model using IV method

Remember the function stsls?

Spatial Lag Model using IV method

Remember the function stsls?

```
> lag_IV <- stsls(log(CMEDV) ~CRIM + CHAS + RM +  AGE +
+ log(RAD) + TAX + PTRATIO + log(LSTAT),data=boston.c,
+ listw = Boston.listw)
> summary(lag_IV)

Call:
stsls(formula = log(CMEDV) ~ CRIM + CHAS + RM + AGE + log(RAD) +
    TAX + PTRATIO + log(LSTAT), data = boston.c, listw = Boston.listw)

Residuals:
      Min        1Q    Median        3Q       Max
-0.510851 -0.085687 -0.012095  0.083762  0.793087

Coefficients:
               Estimate  Std. Error  t value  Pr(>|t|)
Rho          3.9130e-01  4.3063e-02   9.0868 < 2.2e-16
(Intercept)  2.3461e+00  2.3266e-01  10.0836 < 2.2e-16
CRIM        -7.1800e-03  1.0796e-03  -6.6504 2.923e-11
CHAS1        3.1342e-02  2.8456e-02   1.1014 0.2707216
RM           6.8380e-02  1.3767e-02   4.9669 6.802e-07
AGE          1.1829e-03  3.3478e-04   3.5335 0.0004100
log(RAD)     6.5166e-02  1.5371e-02   4.2396 2.239e-05
TAX         -3.3453e-04  8.6321e-05  -3.8754 0.0001064
PTRATIO     -1.4230e-02  3.8733e-03  -3.6738 0.0002390
log(LSTAT)  -2.9932e-01  2.4610e-02 -12.1626 < 2.2e-16

Residual variance (sigma squared): 0.023634, (sigma: 0.15373)
```

Estimate a Spatial Error Model - ML

Remember the errorsarlm function?

Estimate a Spatial Error Model - ML

Remember the errorsarlm function?

```
> error_ml<-errorsarlm(log(CMEDV) ~CRIM + CHAS + RM +  AGE +
+ log(RAD) + TAX + PTRATIO + log(LSTAT),data=boston.c,
+ listw = Boston.listw, method="eigen")
> summary(error_ml)
Call:
errorsarlm(formula = log(CMEDV) ~ CRIM + CHAS + RM + AGE + log(RAD) +
    TAX + PTRATIO + log(LSTAT), data = boston.c, listw = Boston.listw,
    method = "eigen")

Residuals:
       Min         1Q     Median         3Q        Max
-0.6047294 -0.0734572 -0.0027824  0.0800144  0.6329991

Type: error
Coefficients: (asymptotic standard errors)
               Estimate  Std. Error  z value  Pr(>|z|)
(Intercept)  3.72327104  0.15050080  24.7392 < 2.2e-16
CRIM        -0.00546888  0.00096789  -5.6503 1.602e-08
CHAS1       -0.02865598  0.02851375  -1.0050 0.3149025
RM           0.08504883  0.01357816   6.2636 3.761e-10
AGE         -0.00044718  0.00046974  -0.9520 0.3411134
log(RAD)     0.06350036  0.02089916   3.0384 0.0023782
TAX         -0.00055632  0.00010958  -5.0767 3.841e-07
PTRATIO     -0.01853791  0.00553272  -3.3506 0.0008064
log(LSTAT)  -0.30428476  0.02248880 -13.5305 < 2.2e-16

Lambda: 0.7225, LR test value: 255.31, p-value: < 2.22e-16
Asymptotic standard error: 0.031185
    z-value: 23.168, p-value: < 2.22e-16
Wald statistic: 536.74, p-value: < 2.22e-16

Log likelihood: 248.2429 for error model
ML residual variance (sigma squared): 0.018415, (sigma: 0.1357)
Number of observations: 506
Number of parameters estimated: 11
AIC: -474.49, (AIC for lm: -221.17)
```

Estimate a Spatial Error Model - GM

Remember the GMerrorsar function?

Estimate a Spatial Error Model - GM

Remember the GMerrorsar function?

```
> error_gm<-GMerrorsar(log(CMEDV) ~CRIM + CHAS + RM +  AGE +
+ log(RAD) + TAX + PTRATIO + log(LSTAT),data=boston.c,
+ listw = Boston.listw)
> summary(error_gm)

Call:
GMerrorsar(formula = log(CMEDV) ~ CRIM + CHAS + RM + AGE + log(RAD) +
    TAX + PTRATIO + log(LSTAT), data = boston.c, listw = Boston.listw)

Residuals:
       Min         1Q     Median         3Q        Max
-0.7483190 -0.1118763 -0.0090407  0.1106119  0.9773213

Type: GM SAR estimator
Coefficients: (GM standard errors)
               Estimate  Std. Error  z value  Pr(>|z|)
(Intercept)  3.8120e+00  1.5976e-01  23.8607 < 2.2e-16
CRIM        -6.5641e-03  1.0720e-03  -6.1233 9.165e-10
CHAS1       -2.6048e-03  3.1246e-02  -0.0834 0.9335634
RM           8.2962e-02  1.5074e-02   5.5037 3.719e-08
AGE         -6.6566e-05  4.7786e-04  -0.1393 0.8892130
log(RAD)     6.6159e-02  2.1295e-02   3.1068 0.0018913
TAX         -5.5470e-04  1.1360e-04  -4.8828 1.046e-06
PTRATIO     -2.0842e-02  5.5513e-03  -3.7544 0.0001738
log(LSTAT)  -3.3048e-01  2.4234e-02 -13.6372 < 2.2e-16

Lambda: 0.57506 (standard error): 0.068782 (z-value): 8.3606
Residual variance (sigma squared): 0.022746, (sigma: 0.15082)
GM argmin sigma squared: 0.023157
Number of observations: 506
Number of parameters estimated: 11
```

Estimate a Spatial Error Model - GM

Remember the spatial test results? It might be better to see the combination of Spatial Lag and Error models.

Estimate a Spatial Error Model - GM

Remember the spatial test results? It might be better to see the combination of Spatial Lag and Error models.

```
> sacsarOLS.Boston <- sacsarlm(log(CMEDV) ~CRIM + CHAS + RM +  AGE +
+ log(RAD) + TAX + PTRATIO + log(LSTAT),data=boston.c,
+ listw = Boston.listw)
> summary(sacsarOLS.Boston)
```

```
Call:
sacsarlm(formula = log(CMEDV) ~ CRIM + CHAS + RM + AGE + log(RAD) +
    TAX + PTRATIO + log(LSTAT), data = boston.c, listw = Boston.listw)

Residuals:
       Min         1Q     Median         3Q        Max
-0.6091799 -0.0744564 -0.0078916  0.0815700  0.7004352

Type: sac
Coefficients: (asymptotic standard errors)
               Estimate  Std. Error  z value  Pr(>|z|)
(Intercept)  3.21729868  0.24853263  12.9452 < 2.2e-16
CRIM        -0.00580433  0.00099295  -5.8456 5.049e-09
CHAS1       -0.01737403  0.02894695  -0.6002 0.5483713
RM           0.08602878  0.01396193   6.1617 7.198e-10
AGE         -0.00001224  0.00045688  -0.0268 0.9786269
log(RAD)     0.07122072  0.02007977   3.5469 0.0003898
TAX         -0.00052315  0.00010739  -4.8715 1.108e-06
PTRATIO     -0.01903641  0.00524835  -3.6271 0.0002866
log(LSTAT)  -0.30856250  0.02290598 -13.4708 < 2.2e-16

Rho: 0.152
Asymptotic standard error: 0.052726
    z-value: 2.8828, p-value: 0.003941
Lambda: 0.60375
Asymptotic standard error: 0.05196
    z-value: 11.619, p-value: < 2.22e-16

LR test value: 258.77, p-value: < 2.22e-16

Log likelihood: 249.9701 for sac model
ML residual variance (sigma squared): 0.019377, (sigma: 0.1392)
Number of observations: 506
```

Dealing with Heterosckedasticity in Spatial Models

Getting into Panel Models: Fixed vs Random

Consider a linear panel model:

$$y_{it} = X_{it}\beta + u_{it} \tag{1}$$

Here the error term u_{it} is our concern.
If we believe that the unobservable factors effect the ith value of y in time t then we have to deal with that.
Most of the panel data applications assume that the composite error term vit follows a one-way error structure,

$$u_{it} = \mu_i + \epsilon_{it}$$

that has two components, μ_i specific to countries that does not change over time and u_{it} that changes both over time and for countries.

Getting into Panel Models: Fixed vs Random

Consider a linear panel model:

$$y_{it} = X_{it}\beta + u_{it} \quad (1)$$

Here the error term u_{it} is our concern.
If we believe that the unobservable factors effect the ith value of y in time t then we have to deal with that.
Most of the panel data applications assume that the composite error term vit follows a one-way error structure,

$$u_{it} = \mu_i + \epsilon_{it}$$

that has two components, μ_i specific to countries that does not change over time and u_{it} that changes both over time and for countries.

In order to find if the spatial specific effects (μ_i) are significant and therefore the estimation of the model with the pooled Ordinary Least Squares (OLS) estimates would be biased and inconsistent, joint significance of the country specific factors with F-test should be applied.

Getting into Panel Models: Fixed vs Random

- This test basically compares the sum square errors of the fixed effects and OLS methods, and if a significant F is obtained then the test reveals that the country specific effects are significant (Baltagi, 2011).

Getting into Panel Models: Fixed vs Random

- This test basically compares the sum square errors of the fixed effects and OLS methods, and if a significant F is obtained then the test reveals that the country specific effects are significant (Baltagi, 2011).
- Though at this point, another concern arises on how to treat these effects. Since it is found that there are unobserved country specific factors which causes the error to be autocorrelated, the estimation of (1) by the ordinary least squares becomes impossible (Wooldridge, 2002).

Getting into Panel Models: Fixed vs Random

- This test basically compares the sum square errors of the fixed effects and OLS methods, and if a significant F is obtained then the test reveals that the country specific effects are significant (Baltagi, 2011).
- Though at this point, another concern arises on how to treat these effects. Since it is found that there are unobserved country specific factors which causes the error to be autocorrelated, the estimation of (1) by the ordinary least squares becomes impossible (Wooldridge, 2002).
- Therefore there are methods developed for panel data. These methods analyse the u_{it} composite error term by assuming that the one-way error structure is either fixed or random (Frees, 2004, Wooldridge, 2002).

Getting into Panel Models: Fixed vs Random

- This test basically compares the sum square errors of the fixed effects and OLS methods, and if a significant F is obtained then the test reveals that the country specific effects are significant (Baltagi, 2011).
- Though at this point, another concern arises on how to treat these effects. Since it is found that there are unobserved country specific factors which causes the error to be autocorrelated, the estimation of (1) by the ordinary least squares becomes impossible (Wooldridge, 2002).
- Therefore there are methods developed for panel data. These methods analyse the u_{it} composite error term by assuming that the one-way error structure is either fixed or random (Frees, 2004, Wooldridge, 2002).
- The decision on which method to use is the most important thing in panel data analysis (Matyas, Sevestre, 1996) and it depends on the existence of correlation between the explanatory variables and the unobservable cross sectional specific factors (Arellano, 2003, Wooldridge, 2002). There is no restriction about zero correlation between the explanatory variables and the unobservable cross sectional specific factors (μ_i) in the fixed effects models. However, random effects estimates are based on zero correlation assumption. As a result, if the correlations between the explanatory variables and the unobservable cross sectional specific factors (μ_i) are found to be high then one should hesitate in estimating the panel data with random effects since the basic assumption would be violated.

Getting into Panel Models: Fixed vs Random

- There is also a Hausman test that compares the less efficient fixed effects model with the more efficient random effects model under the null of no significant difference. The test tries to find if the more efficient random effects model also gives consistent results and if the null is not rejected then the more efficient random effects model is preferred.

Getting into Panel Models: Fixed vs Random

- There is also a Hausman test that compares the less efficient fixed effects model with the more efficient random effects model under the null of no significant difference. The test tries to find if the more efficient random effects model also gives consistent results and if the null is not rejected then the more efficient random effects model is preferred.
- On the other hand, if the test finds that there is significant difference between the models, then it is more appropriate to use fixed effects models. Though it has to be kept in mind that Hausman test is not a test to make a choice between fixed or random. There is more to be kept in mind. If the data set includes all of the cross-sectional units rather than a randomly chosen sample from the large population, it is more appropriate to obtain the fixed effects model estimates.

Getting into Panel Models: Fixed vs Random

- There is also a Hausman test that compares the less efficient fixed effects model with the more efficient random effects model under the null of no significant difference. The test tries to find if the more efficient random effects model also gives consistent results and if the null is not rejected then the more efficient random effects model is preferred.
- On the other hand, if the test finds that there is significant difference between the models, then it is more appropriate to use fixed effects models. Though it has to be kept in mind that Hausman test is not a test to make a choice between fixed or random. There is more to be kept in mind. If the data set includes all of the cross-sectional units rather than a randomly chosen sample from the large population, it is more appropriate to obtain the fixed effects model estimates.
- Once it is decided to estimate a fixed effects model, it means that μ_i is assumed to be fixed parameters to be estimated and therefore least squares dummy variable estimation approach or within group transformations are applied, whereas if it is decided for a random effects model, then μ_i is assumed to be a random component and generalized least squares estimation approach is applied which is nothing more than applying OLS to the transformed variables with the inverse of the variance-covariance matrix of the composite error terms (Baltagi, 2001, Bhargava, Franzini & Narendranathan, 1982).

Tests in Panel Data Analysis

1. Tests of poolability
   tests the hypothesis that the same coeffcients apply to each individual. It is a standard F test, based on the comparison of a model obtained for the full sample and a model based on the estimation of an equation for each individual.

Tests in Panel Data Analysis

1. Tests of poolability
   tests the hypothesis that the same coeffcients apply to each individual. It is a standard F test, based on the comparison of a model obtained for the full sample and a model based on the estimation of an equation for each individual.
2. Tests for individual and time effects
   This is Lagrange multiplier tests of individual or/and time effects based on the results of the pooling model.

Tests in Panel Data Analysis

1. Tests of poolability
   tests the hypothesis that the same coeffcients apply to each individual. It is a standard F test, based on the comparison of a model obtained for the full sample and a model based on the estimation of an equation for each individual.
2. Tests for individual and time effects
   This is Lagrange multiplier tests of individual or/and time effects based on the results of the pooling model.
3. Hausman test
   Hausman test which is based on the comparison of two sets of estimates (see Hausman 1978). A classical application of the Hausman test for panel data is to compare the fixed and the random effects models.

Tests in Panel Data Analysis

1. Tests of poolability
   tests the hypothesis that the same coeffcients apply to each individual. It is a standard F test, based on the comparison of a model obtained for the full sample and a model based on the estimation of an equation for each individual.
2. Tests for individual and time effects
   This is Lagrange multiplier tests of individual or/and time effects based on the results of the pooling model.
3. Hausman test
   Hausman test which is based on the comparison of two sets of estimates (see Hausman 1978). A classical application of the Hausman test for panel data is to compare the fixed and the random effects models.
4. Tests for cross-sectional dependence

Panel Estimation in R: Fixed vs Random

We will use plm package in developed by Croissant and Millo.

```
> library(plm)
```

Panel Estimation in R: Fixed vs Random

We will use plm package in developed by Croissant and Millo.

```
> library(plm)
```

plm provides four functions for estimation:

- plm: estimation of the basic panel models, i.e., within, between and random effect models. Models are estimated using the lm function to transformed data,
- pvcm: estimation of models with variable coefficients,

```
> data(Grunfeld)
> fm <- inv ~ value + capital
```

Panel Estimation in R: Fixed vs Random

We will use plm package in developed by Croissant and Millo.

```
> library(plm)
```

plm provides four functions for estimation:

- plm: estimation of the basic panel models, i.e., within, between and random effect models. Models are estimated using the lm function to transformed data,
- pvcm: estimation of models with variable coefficients,
- pgmm: estimation of generalized method of moments models,

```
> data(Grunfeld)
> fm <- inv ~ value + capital
```

A panelmodel object contains:

coefficients, residuals, fitted.values, vcov, df.residual and call.

Panel Estimation in R: Fixed vs Random

We will use plm package in developed by Croissant and Millo.

```
> library(plm)
```

plm provides four functions for estimation:

- plm: estimation of the basic panel models, i.e., within, between and random effect models. Models are estimated using the lm function to transformed data,
- pvcm: estimation of models with variable coefficients,
- pgmm: estimation of generalized method of moments models,
- pggls: estimation of general feasible generalized least squares models.

```
> data(Grunfeld)
> fm <- inv ~ value + capital
```

A panelmodel object contains:

coefficients, residuals, fitted.values, vcov, df.residual and call.

Panel Estimation in R: Fixed vs Random

Several models can be estimated with plm by
lling the model argument:

- the fixed effects model (within),

Panel Estimation in R: Fixed vs Random

Several models can be estimated with plm by
lling the model argument:

- the fixed effects model (within),
- the pooling model (pooling),

Panel Estimation in R: Fixed vs Random

Several models can be estimated with plm by
lling the model argument:

- the fixed effects model (within),
- the pooling model (pooling),
- the first-difference model (fd),

Panel Estimation in R: Fixed vs Random

Several models can be estimated with plm by
lling the model argument:

- the fixed effects model (within),
- the pooling model (pooling),
- the first-difference model (fd),
- the between model (between),

Panel Estimation in R: Fixed vs Random

Several models can be estimated with plm by
lling the model argument:

- the fixed effects model (within),
- the pooling model (pooling),
- the first-difference model (fd),
- the between model (between),
- the error components model (random).

Grunfeld data: Fixed Effects

For fixed effects:

```
> grun.fe <- plm(fm, data = Grunfeld, model = "within")
```

Grunfeld data: Fixed Effects

For fixed effects:

```
> grun.fe <- plm(fm, data = Grunfeld, model = "within")
```

Fixed effects may be extracted easily using fixef function:

```
> fixef(grun.fe)
```

```
         1          2          3          4          5
-70.296717 101.905814 -235.571841 -27.809295 -114.616813
         6          7          8          9         10
-23.161295 -66.553474 -57.545657 -87.222272  -6.567844
```

And you can summarize the fixed effects:

```
> summary(fixef(grun.fe))
```

```
    Estimate Std. Error t-value  Pr(>|t|)
1   -70.2967    49.7080 -1.4142  0.15730
2   101.9058    24.9383  4.0863 4.383e-05 ***
3  -235.5718    24.4316 -9.6421 < 2.2e-16 ***
4   -27.8093    14.0778 -1.9754  0.04822 *
5  -114.6168    14.1654 -8.0913 6.661e-16 ***
6   -23.1613    12.6687 -1.8282  0.06752 .
7   -66.5535    12.8430 -5.1821 2.194e-07 ***
8   -57.5457    13.9931 -4.1124 3.915e-05 ***
9   -87.2223    12.8919 -6.7657 1.327e-11 ***
10   -6.5678    11.8269 -0.5553  0.57867
---
Signif. codes:  0 Ś***Š 0.001 Ś**Š 0.01 Ś*Š 0.05 Ś.Š 0.1 Ś Š 1
```

Grunfeld data: Random Effects

For random effects:

```
> grun.re <- plm(fm, data = Grunfeld, model = "random")
```

Grunfeld data: Random Effects

For random effects:

```
> grun.re <- plm(fm, data = Grunfeld, model = "random")
```

The random effect model is obtained as a linear estimation on quasidemeaned data. The parameter of this transformation is obtained using preliminary estimations. Four estimators of this parameter are available, depending on the value of the argument random.method: swar, walhus, amemiya,nerlove. Default is swar.

Grunfeld data: Random Effects

For random effects:

```
> grun.re <- plm(fm, data = Grunfeld, model = "random")
```

The random effect model is obtained as a linear estimation on quasidemeaned data. The parameter of this transformation is obtained using preliminary estimations. Four estimators of this parameter are available, depending on the value of the argument random.method: swar, walhus, amemiya,nerlove. Default is swar.

```
> grun.re <- plm(fm, data = Grunfeld, model = "random", random.method="amemiya")
```

Grunfeld data: Two-Way Effects - both individual and time effects

If you want to estimate a time-effect model, then use the effect option:

Grunfeld data: Two-Way Effects - both individual and time effects

If you want to estimate a time-effect model, then use the effect option:

```
> grun.time<- plm(fm, data = Grunfeld, effect="time", model = "random",
+ random.method="amemiya")
```

Grunfeld data: Two-Way Effects - both individual and time effects

If you want to estimate a time-effect model, then use the effect option:

```
> grun.time<- plm(fm, data = Grunfeld, effect="time", model = "random",
+ random.method="amemiya")
```

If you want both effects:

```
> grun.twoways<- plm(fm, data = Grunfeld, effect="twoways", model = "random",
+ random.method="amemiya")
```

Grunfeld data - Tests: Poolability

```
> pooltest(fm, data = Grunfeld, model = "within")

F statistic

data:  fm
F = 5.7805, df1 = 18, df2 = 170, p-value = 1.219e-10
alternative hypothesis: unstability
```

Grunfeld data - Tests: Individual and Time Effects

plmtest implements Lagrange multiplier tests of individual or/and time effects based on the results of the pooling model. Its main argument is a plm object (the result of a pooling model) or a formula.

Grunfeld data - Tests: Individual and Time Effects

plmtest implements Lagrange multiplier tests of individual or/and time effects based on the results of the pooling model. Its main argument is a plm object (the result of a pooling model) or a formula.
The effects tested are indicated with the effect argument (one of individual, time or twoways).
To test for the presence of both individual and time effects:

```
> g <- plm(fm, data = Grunfeld, model = "pooling")
> plmtest(g, effect = "twoways", type = "ghm")
```

```
Lagrange Multiplier Test - two-ways effects (Gourieroux,
Holly and Monfort)

data:  fm
chisq = 798.1615, df = 2, p-value < 2.2e-16
alternative hypothesis: significant effects
```

Grunfeld data - Tests: Individual and Time Effects

pFtest computes F tests of effects based on the comparison of the within and the pooling models. Its main arguments are either two plm objects (the results of a pooling and a within model) or a formula.

Grunfeld data - Tests: Individual and Time Effects

pFtest computes F tests of effects based on the comparison of the within and the pooling models. Its main arguments are either two plm objects (the results of a pooling and a within model) or a formula.

```
> gw <- plm(fm, data = Grunfeld, effect = "twoways",model = "within")
> gp <- plm(fm, data = Grunfeld, model = "pooling")
> pFtest(gw, gp)

F test for twoways effects

data:  fm
F = 17.4031, df1 = 28, df2 = 169, p-value < 2.2e-16
alternative hypothesis: significant effects
```

Grunfeld data - Tests: Hausman Test

Hausman test compares the fixed and the random effects: models:

Grunfeld data - Tests: Hausman Test

Hausman test compares the fixed and the random effects: models:

```
> gw <- plm(inv ~ value + capital, data = Grunfeld, model = "within")
> gr <- plm(inv ~ value + capital, data = Grunfeld, model = "random")
> phtest(gw, gr)
```

```
Hausman Test

data:  inv ~ value + capital
chisq = 2.3304, df = 2, p-value = 0.3119
alternative hypothesis: one model is inconsistent
```

Spatial Panel Estimation

The spatial lag model (SAR) posits that the dependent variable depends on the dependent variable observed in neighbouring units and on a set of observed local characteristics.

$$Y_{it} = \rho \sum_{j=1}^{N} w_{ij} y_{jt} + \mathbf{x}_{it} \boldsymbol{\beta} + \mu_i + \epsilon_{it} \tag{2}$$

Spatial Panel Estimation

The spatial lag model (SAR) posits that the dependent variable depends on the dependent variable observed in neighbouring units and on a set of observed local characteristics.

$$Y_{it} = \rho \sum_{j=1}^{N} w_{ij} y_{jt} + \mathbf{x}_{it} \boldsymbol{\beta} + \mu_i + \epsilon_{it} \tag{2}$$

The spatial error model (SEM), on the other hand, posits that the dependent variable depends on a set of observed local characteristics and that the error terms are correlated across space

$$Y_{it} = \mathbf{x}_{it} \boldsymbol{\beta} + \mu_i + \epsilon_{it} \tag{3}$$

$$\epsilon_{it} = \lambda \sum_{j=1}^{N} w_{ij} \epsilon_{it} + \nu_{it} \tag{4}$$

Here the spatial specific effects (μ_i) are approached as fixed or random.

Spatial Panel Estimation - Fixed Effect Spatial Lag

According to Anselin et al. (2006), the extension of the fixed effects model with a spatially lagged dependent variable raises two complications. First, the endogeneity of $\sum_j w_{ij} y_{jt}$ violates the assumption of the standard regression model that $E[(\sum_j w_{ij} y_{jt})\epsilon_{it}] = 0$. In model estimation, this simultaneity must be accounted for. Second, the spatial dependence among the observations at each point in time may affect the estimation of the fixed effects.

Spatial Panel Estimation - Fixed Effect Spatial Lag

According to Anselin et al. (2006), the extension of the fixed effects model with a spatially lagged dependent variable raises two complications. First, the endogeneity of $\sum_j w_{ij} y_{jt}$ violates the assumption of the standard regression model that $E[(\sum_j w_{ij} y_{jt})\epsilon_{it}] = 0$. In model estimation, this simultaneity must be accounted for. Second, the spatial dependence among the observations at each point in time may affect the estimation of the fixed effects.

Two main approaches have been suggested in the literature to estimate models that include spatial interaction effects. One is based on the maximum likelihood (ML) principle and the other on instrumental variables or generalized method of moments (IV/GMM) techniques.

Using splm - Munnell's productivity model

Munnell (1990), Public capital productivity: Does public capital (roads, water facilities, public buildings and structures) help growth? (Example 3 in Baltagi) 48 US states, annual data 1970-1986. Production function:
$log(gsp) = \alpha + \beta_1 log(pcap) + \beta_2 log(pc) + \beta_3 log(emp) + \beta_4 unemp$

```
> library(splm)
> data(Produc, package="Ecdat")
> data(usaww)
> fm <- log(gsp)~log(pcap)+log(pc)+log(emp)+unemp
```

Using splm - Munnell's productivity model

Munnell (1990), Public capital productivity: Does public capital (roads, water facilities, public buildings and structures) help growth? (Example 3 in Baltagi) 48 US states, annual data 1970-1986. Production function:

$log(gsp) = \alpha + \beta_1 log(pcap) + \beta_2 log(pc) + \beta_3 log(emp) + \beta_4 unemp$

```
> library(splm)
> data(Produc, package="Ecdat")
> data(usaww)
> fm <- log(gsp)~log(pcap)+log(pc)+log(emp)+unemp
```

Now check the usaww and Produc... The weights matrix should be a 48X48 matrix. Attention: It has already been row-standardized. You could have done it with the mat2listw function by setting the style to "w"

Create weights matrix

Since the weights matrix is in a matrix format, we have to transform it into a listw format.

Create weights matrix

Since the weights matrix is in a matrix format, we have to transform it into a listw format.

```
> usalw <- mat2listw(usaww)
```

Pooling SAR

Pooling SAR

```
> sarpool <- spml(formula = fm, data = Produc, listw = usalw,
+ model = "pooling", spatial.error = "none", lag = TRUE)
> summary(sarpool)
Spatial panel random effects ML model

Call:
spml(formula = fm, data = Produc, listw = usalw, model = "pooling",
    lag = TRUE, spatial.error = "none")

Residuals:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-0.2530 -0.0825 -0.0219 -0.0218  0.0285  0.3320

Spatial autoregressive coefficient:
         Estimate Std. Error t-value Pr(>|t|)
lambda -0.0020763  0.0057539 -0.3609   0.7182

Coefficients:
              Estimate Std. Error t-value  Pr(>|t|)
(Intercept)  1.6669444  0.0574063 29.0377 < 2.2e-16 ***
log(pcap)    0.1533182  0.0170999  8.9660 < 2.2e-16 ***
log(pc)      0.3091957  0.0102397 30.1958 < 2.2e-16 ***
log(emp)     0.5958931  0.0137043 43.4823 < 2.2e-16 ***
unemp       -0.0066072  0.0014119 -4.6796 2.875e-06 ***
```

Pooling SEM

Pooling SEM

```
> sempool <- spml(formula = fm, data = Produc, listw = usalw,
+ model = "pooling",spatial.error = "b", lag = FALSE)
> summary(sempool)
Spatial panel random effects ML model

Call:
spml(formula = fm, data = Produc, listw = usalw, model = "pooling",
    lag = FALSE, spatial.error = "b")

Residuals:
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-0.22800 -0.06940 -0.00304 -0.00327  0.05060  0.30500

Error variance parameters:
    Estimate Std. Error t-value  Pr(>|t|)
rho 0.520843   0.037443   13.91 < 2.2e-16 ***

Coefficients:
              Estimate Std. Error t-value  Pr(>|t|)
(Intercept)  1.4055761  0.0579229 24.2663 < 2.2e-16 ***
log(pcap)    0.1417134  0.0164206  8.6302 < 2.2e-16 ***
log(pc)      0.3676667  0.0109693 33.5178 < 2.2e-16 ***
log(emp)     0.5602226  0.0143948 38.9185 < 2.2e-16 ***
unemp       -0.0086340  0.0017268 -5.0001 5.732e-07 ***
```

SEM - RE

SEM - RE

```
> semre <- spml(formula = fm, data = Produc, listw = usalw,
+ model = "random",spatial.error = "b", lag = FALSE)
> summary(semre)
Spatial panel random effects ML model

Call:
spml(formula = fm, data = Produc, listw = usalw, model = "random",
    lag = FALSE, spatial.error = "b")

Residuals:
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-0.26700 -0.06050 -0.00676 -0.00003  0.05360  0.46600

Error variance parameters:
    Estimate Std. Error t-value  Pr(>|t|)
phi  7.49517    1.73519  4.3195 1.564e-05 ***
rho  0.53888    0.03371 15.9855 < 2.2e-16 ***

Coefficients:
              Estimate Std. Error t-value Pr(>|t|)
(Intercept)  2.3868273  0.1393798 17.1246  < 2e-16 ***
log(pcap)    0.0424139  0.0222037  1.9102  0.05611 .
log(pc)      0.2418396  0.0202892 11.9196  < 2e-16 ***
log(emp)     0.7423454  0.0244061 30.4164  < 2e-16 ***
```

SEM - FE

```
> semfe <- spml(formula = fm, data = Produc, listw = usalw,
+ model = "within",effect = "individual",spatial.error = "b", lag = FALSE)
> summary(semfe)

Spatial panel fixed effects error model

Call:
spml(formula = fm, data = Produc, listw = usalw, model = "within",
    effect = "individual", lag = FALSE, spatial.error = "b")

Residuals:
   Min. 1st Qu.  Median 3rd Qu.    Max.
-0.1250 -0.0238 -0.0035  0.0171  0.1880

Coefficients:
            Estimate Std. Error t-value Pr(>|t|)
rho        0.5574013  0.0330749 16.8527  < 2e-16 ***
log(pcap)  0.0051438  0.0250109  0.2057  0.83705
log(pc)    0.2053026  0.0231427  8.8712  < 2e-16 ***
log(emp)   0.7822540  0.0278057 28.1328  < 2e-16 ***
unemp     -0.0022317  0.0010709 -2.0839  0.03717 *
---
Signif. codes:  0 Ś***Š 0.001 Ś**Š 0.01 Ś*Š 0.05 Ś.Š 0.1 Ś Š 1
```

SEM - The Effects

```
> eff <- effects(semfe)
> eff
Intercept:
             Estimate Std. Error t-value  Pr(>|t|)
(Intercept)  2.84695    0.14324  19.876 < 2.2e-16 ***


Spatial fixed effects:
     Estimate Std. Error t-value Pr(>|t|)
1  -0.1393449  0.1442142 -0.9662  0.33393
2   0.0042234  0.1435461  0.0294  0.97653
3  -0.0993829  0.1369773 -0.7255  0.46812
4   0.2059001  0.1648750  1.2488  0.21173
5   0.0422873  0.1417953  0.2982  0.76553
6   0.0912363  0.1410148  0.6470  0.51763
7  -0.0168266  0.1325054 -0.1270  0.89895
8   0.0084710  0.1501192  0.0564  0.95500
9  -0.0791192  0.1452860 -0.5446  0.58605
10 -0.0556094  0.1308325 -0.4250  0.67081
11  0.0854043  0.1557281  0.5484  0.58340
12 -0.0570390  0.1455946 -0.3918  0.69523
13 -0.0121360  0.1468313 -0.0827  0.93413
14  0.0088455  0.1454997  0.0608  0.95152
15  0.0459446  0.1466921  0.3132  0.75413
```

SAR - RE

```
> sarre <- spml(formula = fm, data = Produc, listw = usalw,
+ model = "random",spatial.error = "none", lag = TRUE)
> summary(sarre)
Spatial panel random effects ML model

Call:
spml(formula = fm, data = Produc, listw = usalw, model = "random",
    lag = TRUE, spatial.error = "none")

Residuals:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1.38    1.57    1.70    1.70    1.80    2.13

Error variance parameters:
    Estimate Std. Error t-value Pr(>|t|)
phi  21.3175     8.3017  2.5678  0.01023 *

Spatial autoregressive coefficient:
       Estimate Std. Error t-value  Pr(>|t|)
lambda 0.161615   0.029099   5.554 2.793e-08 ***

Coefficients:
               Estimate  Std. Error t-value  Pr(>|t|)
(Intercept)  1.65814995  0.15071855 11.0016 < 2.2e-16 ***
```

SAR - FE

```
> sarfe <- spml(formula = fm, data = Produc, listw = usalw, model = "within",
+ effect = "individual", spatial.error = "none", lag = TRUE)
> summary(sarfe)

Spatial panel fixed effects lag model

Call:
spml(formula = fm, data = Produc, listw = usalw, model = "within",
    effect = "individual", lag = TRUE, spatial.error = "none")

Residuals:
   Min. 1st Qu.  Median 3rd Qu.    Max.
-0.1510 -0.0191 -0.0025  0.0159  0.1770

Coefficients:
            Estimate Std. Error t-value  Pr(>|t|)
lambda     0.2746887  0.0235164 11.6807 < 2.2e-16 ***
log(pcap) -0.0465819  0.0254425 -1.8309   0.06712 .
log(pc)    0.1874325  0.0230441  8.1336 4.166e-16 ***
log(emp)   0.6250902  0.0297044 21.0437 < 2.2e-16 ***
unemp     -0.0044816  0.0008653 -5.1792 2.228e-07 ***
---
Signif. codes:  0 Ś***Š 0.001 Ś**Š 0.01 Ś*Š 0.05 Ś.Š 0.1 Ś Š 1
```

SAR - SEM - RE

```
> sararremod <- spml(formula = fm, data = Produc, index = NULL,
+ listw = usalw,  model = "random", lag = TRUE, spatial.error = "b")
> summary(sararremod)
Spatial panel random effects ML model

Call:
spml(formula = fm, data = Produc, index = NULL, listw = usalw,
    model = "random", lag = TRUE, spatial.error = "b")

Residuals:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
-0.2480 -0.0411  0.0123  0.0191  0.0726  0.4840

Error variance parameters:
    Estimate Std. Error t-value  Pr(>|t|)
phi 7.530808   1.748372  4.3073 1.652e-05 ***
rho 0.536835   0.034098 15.7439 < 2.2e-16 ***

Spatial autoregressive coefficient:
        Estimate Std. Error t-value Pr(>|t|)
lambda 0.0018174  0.0045022  0.4037   0.6864

Coefficients:
              Estimate Std. Error t-value  Pr(>|t|)
```

SAR - SEM - FE

```
> sararfemod <- spml(formula = fm, data = Produc, index = NULL,
+ listw = usalw, lag = TRUE, spatial.error= "b", model = within,
+ effect = individual, method = eigen, na.action = na.fail,
+ quiet = TRUE, zero.policy = NULL, interval = NULL, tol.solve = 1e-10,
+ control = list(), legacy = FALSE )
> summary(sararfemod)
Spatial panel fixed effects sarar model

Call:
spml(formula = fm, data = Produc, index = NULL, listw = usalw,
    model = "within", effect = "individual", lag = TRUE, spatial.error = "b",
    method = "eigen", na.action = na.fail, quiet = TRUE, zero.policy = NULL,
    interval = NULL, tol.solve = 1e-10, control = list(), legacy = FALSE)

Residuals:
   Min. 1st Qu.  Median 3rd Qu.    Max.
-0.1340 -0.0221 -0.0032  0.0172  0.1750

Coefficients:
            Estimate Std. Error t-value  Pr(>|t|)
rho        0.4553116  0.0501451  9.0799 < 2.2e-16 ***
lambda     0.0885760  0.0297655  2.9758  0.002922 **
log(pcap) -0.0103497  0.0252725 -0.4095  0.682156
log(pc)    0.1905781  0.0230505  8.2678 < 2.2e-16 ***
```

References

1. http://www.eurojournals.com/ejss_18_1_02.pdf
2. http://www.ecomod.org/files/papers/486.pdf
3. http://www.people.fas.harvard.edu/~zhukov/Spatial6.pdf
4. http://www.drs.wisc.edu/documents/articles/curtis/cesoc977-12/W3_W6_W9_GeodaWorkbook.pdf
5. http://dae.unizar.es/docencia/regional/spacestat%20Tutorial.pdf
6. http://www.people.fas.harvard.edu/~zhukov/spatial.html
7. https://geodacenter.asu.edu/system/files/rex1.pdf
8. http://scc.stat.ucla.edu/page_attachments/0000/0094/spatial_R_1_09S.pdf
9. Giuseppe Arbia - Spatial Econometrics book, Springer, 2005.
10. Roger Bivand et all. Applied Spatial Data Analysis with R, Springer, 2008.
11. http://www.jstatsoft.org/v47/i01/paper
12. http://cran.r-project.org/web/packages/splm/splm.pdf
13. http://www.r-project.org/conferences/useR-2009/slides/Millo+Piras.pdf
14. http://cran.r-project.org/web/packages/plm/plm.pdf
15. http://www.jstatsoft.org/v27/i02/paper
16. http://www.spatial-econometrics.com/html/wbook.pdf
17. http://geodacenter.asu.edu/