Some examples of use of **GeoXp** package (version 1.5.0)

T. Laurent[*], A. Ruiz-Gazen[†] and C. Thomas-Agnan[‡]
Toulouse School of Economics

April 21, 2010

1 Description of the basic functionalities

1.1 General principles

Basically, an interactive function[1] of the **GeoXp** package could be called by using one of these following codes:

- `function(sp.obj, name.var,...,options)`, if there is only one variable of interest,
- `function(sp.obj, names.var,...,options)`, if there are several variables of interest,
- `function(sp.obj, name.var, nb.obj...,options)`, if there is one variable of interest and the use of a spatial weight matrix.

The first argument `sp.obj` is a Spatial Class object as defined by R. Bivand in **sp** package. It contains both spatial coordinates and characteristics observed of spatial units.

At this moment, **GeoXp** draws a map, considering spatial units like points: a spatial unit is defined geographically by two scalars x and y. Indeed, for drawing a map, the spatial coordinates of spatial units have been extracted from `sp.obj` by using the function `coordinates`, which could be applied on all Spatial Class object (`SpatialPointsDataFrame`, `SpatialPolygonsDataFrame`, etc).

It also prints a statistical graphic. The variable(s) of interest is given by `name.var` or `names.var`, a (vector of) character (or numeric) which indicates the column(s) of

[*]thibault.laurent@univ-tlse1.fr
[†]ruiz@cict.fr
[‡]cthomas@cict.fr
[1]The non interactive functions included in **GeoXp** correspond to internal or wrap functions

the `sp.obj@data` to be used in the analysis. The `sp.obj@data` is by construction, a `data.frame`.

The ... correspond to specificities of each function (more details in 3). For example, it could indicate the number of bars for histogram, if the y-axis should represent or not the count or the percent for barplot, etc....

Finally, `options` are common to most of the functions with some small specificities by function and described in the following section. Let start with a simple example.

1.2 A very simple usage

This first example has been taken from the example of `histomap` function. We consider a data set included in **GeoXp**, containing price indices of real estate from biggest cities in France in 2008.

```
> data(immob)
> class(immob)

[1] "data.frame"

> names(immob)

 [1] "Nom"                "Code.INSEE"         "Code.region"
 [4] "longitude"          "latitude"           "prix.vente"
 [7] "variation.vente"    "prix.location"      "variation.location"
[10] "rentabilite"
```

As we can see above, this data set is a `data.frame` containing the spatial coordinates of the cities in variables `longitude` and `latitude`. It contains also several variables corresponding to the name of cities, the average price of sell and rent, etc...

The first operation consists in creating a Spatial Object. First, we have to create a `SpatialPoints` object by giving a matrix of 2d with longitude and latitude:

```
> immob.sp = SpatialPoints(cbind(immob$longitude, immob$latitude))
> class(immob.sp)

[1] "SpatialPoints"
attr(,"package")
[1] "sp"
```

Second operation consists in creating a `SpatialPointsDataFrame` by coupling a `SpatialPoints` object with a `data.frame`:

```
> immob.spdf = SpatialPointsDataFrame(immob.sp, immob)
> class(immob.spdf)

[1] "SpatialPointsDataFrame"
attr(,"package")
[1] "sp"
```

Finally, we can call the function `histomap` by giving as first argument, the Spatial Object and as second argument, we give a character (it could have been the number of the column, here the value 6) which corresponds to the name of the variable of interest. The result is the opening of a Tk window and two devices, a device with

number 2 which corresponds to the map and a device with number 3 corresponding to the statistical graph, in this case, the histogram:

```
> histomap(immob.spdf, "prix.vente")
```

As we can see in the Fig. 1, the Tk window contains several buttons that user can click on: user can select a point (`Point` button) or a polygon (`Polygon` button) on the map and can also select a bar on the histogram (`Cell` button). In this example, user could also print bubbles by clicking on `Bubbles` and after choosing a numeric value among the variables included in the Spatial object.

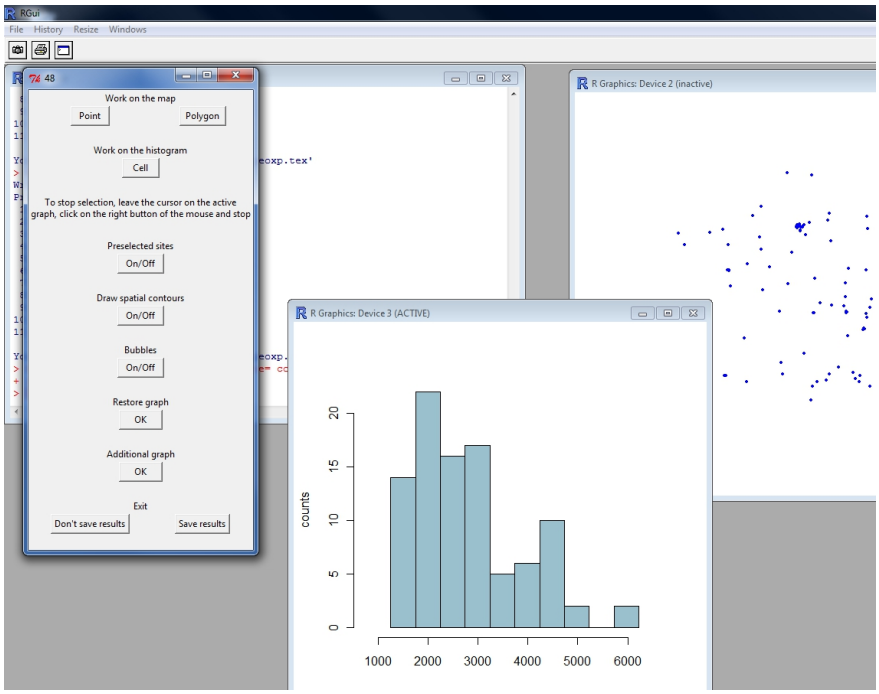

Figure 1: The tk window and the two devices

1.3 To save results

By default, interactive functions don't return results anymore since version 1.5.0. At present, user has to click on `Save results` button to create a global object called `last.select` which is in most the case, a vector of integer containing the number of the last spatial units selected. However, for spatial econometrics functions using spatial weight matrix, `last.select` is a matrix of 2d, because the selection is done on couple of sites (see 3).

1.4 Functions cannot be opened simultaneously

At this moment, user can only open one interactive function. He would have to close the Tk window by clicking on `Save results` or `Don't save results` before opening a new function.

2 Description of the Options

2.1 The options

```
function(sp.obj,...,
names.attr=names(sp.obj), criteria=NULL, carte=NULL,
identify=FALSE, cex.lab=0.8, pch=16, col="lightblue3",
xlab="angle", ylab="absolut magnitude", axes=FALSE,
lablong="", lablat="")
```

Most of these options are common to all the functions. It can differ depending on the function, but the principles stay the same.

- `names.attr`: a vector of character of size the number of variables included in `sp.obj@data`. The option is used for changing the names of variables included in `sp.obj@data`
- criteria: a vector of boolean of size the number of spatial units; it permit to represent preselected sites with a green cross, by clicking on `preselected sites` on the Tk window
- `carte`: in the case where `sp.obj` is a `SpatialPolygonDataFrame`, user will have the opportunity to draw the polygons of Spatial unit by using the `Draw Saptial contours` button in the Tk window. However, if `sp.obj` is a `SpatialPointsDataFrame`, user can give in option `carte`, a matrix with 2 columns for drawing spatial polygonal contours: x and y coordinates of the vertices of the polygon. The functions `polylist2list()` and `spdf2list()` convert some spatial objects (`Polylist` and `SpatialPolygonDataFrame`) into matrix as decribed above to draw a background map.
- `identify`: if TRUE, the names of selected sites will be printed on the map. The names of spatial units correpond to `row.names` of the attribute table `row.names(sp.obj@data)`.
- `cex.lab`: a numeric value, it gives the character size of labels
- `pch`: 16 by default, it gives the symbol for selected points
- `col`: "lightblue3" by default, it gives the color of the bars of histogram, the points of a scatter plot, etc... In the case where the variable of interest is a factor, user could give a vector of colors corresponding to the colors of each level to be printed on the map.
- `xlab`: a character, title for the graphic x-axis

- `ylab`: a character, title for the graphic y-axis
- `axes`: a boolean with TRUE for drawing axes on the map
- `lablong`: a character, name of the x-axis that will be printed on the map
- `lablat`: a character, name of the y-axis that will be printed on the map

2.2 A example with options

We consider the data set `immob` again. We would like to draw as background on the map the spatial contours of the 21 regions in the metropolitan France[2] included in a `shapefile`. For this, we use first the function `readShapePoly` included in **maptools** package, to import the file. Then, we use the function `spdf2list` to convert the `SpatialPolygonsDataFrame` into a matrix of numeric with 2 columns (x and y):

```
> midiP <- readShapePoly(system.file("shapes/region.shp",
+     package = "GeoXp")[1])
> cont_midiP <- spdf2list(midiP[-c(22, 23), ])$poly
```

We also create a vector of boolean which cut approximately the France in two areas, North and South:

```
> criteria <- (immob$latitude > mean(immob$latitude))
```

In the following code, the option `nbcol=15` and `type = "percent"` are specific to function `histomap`. The first one indicates the number of bars to draw and the second the fact that the y-axis of the graphic should represent the percentage of individuals. Notice that the variable of interest corresponds here to the 7th variable of the `sp.obj`, i.e. the variation of sell price observed between 2007 and 2008.

```
> histomap(immob.spdf, 7, nbcol = 15, type = "percent",
+     names.attr = names(immob), criteria = criteria, carte = cont_midiP,
+     identify = TRUE, cex.lab = 0.5, pch = 12, col = "pink",
+     xlab = "variation price", ylab = "percent", axes = TRUE,
+     lablong = "x", lablat = "y")
```

In the Fig. 2, we have represented the two devices after selecting the bars with high values of variable of interest, clicking on `Bubbles` button (and choosing the variable `prix.vente`, average price of sell) and clicking on `Preselected sites` button.

The result on the map and on the graphic is that the selected spatial units are represented in red. Besides on the map, the sites have different sizes depending on the values taken by `prix.vente` and there is a green croice for the cities of the North.

If user click on the `Save results` button, he would obtain the following message and could use the `last.select` object created:

```
[1] "Results have been saved in last.select object"
> last.select
 [1] 12 18 24 31 32 37 39 42 49 67 73 74 79 81 84
```

[2] We have excluded here the regions 22 and 23 which corresponds to the Corse and Andorre

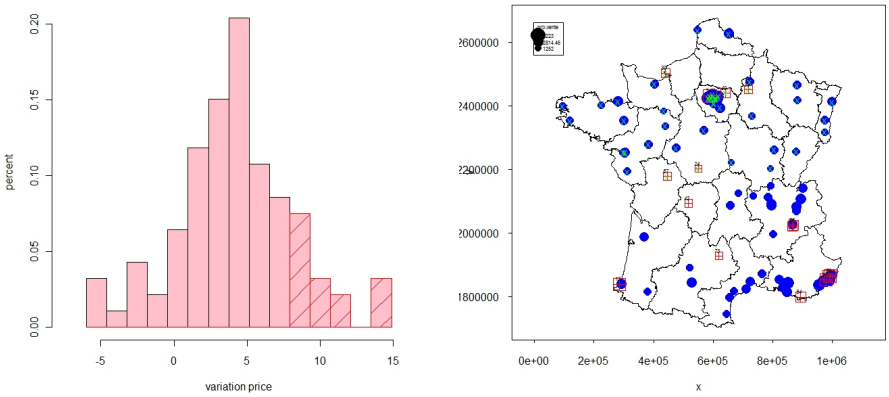

Figure 2: The use of options in histomap function

3 The main functions of GeoXp

We describe here succinctly the statistical graphic, the specific options and the dependencies with other packages.

3.1 Functions with one variable of interest

- `angleplotmap`: absolute difference between the value of `name.var` at two sites as a function of the angle between vector $\overrightarrow{s_i s_j}$ and the x-axis. Specific option is `quantiles`, for drawing a conditionnal quantile curve.
- `barmap`: bar plot (vertical bar) of the given factor variable `name.var`. Specific options are `type` to give the value in the y-axis (count or percent) and `names.arg` to give the names of levels of `name.var`.
- `boxplotmap`: boxplot of the given variable *name.var*.
- `densitymap`: kernel density estimates of the variable `name.var` with `bkde` function included in **KernSmooth** package. Specific option is `kernel` for the choice of kernel.
- `driftmap`: device divided into 2 rows and 2 columns which contains : (cell 1) the selected sites divided into m rows and q columns (m and q are selected with the tk window), (cell 2) a scatter plot with `coordinates(sp.obj)[,2]` in x-axis and the mean and median of `name.var` calculated for the m rows in y-axis, (cell 3) a scatter plot with the mean and median of `nam.var` calculated for the q columns in x-axis and `coordinates(sp.obj)[,1]` in y-axis and (cell 4) a legend indicating the direction of the North. Specific options are `name.var`,

`interpol=TRUE`, `nuage=TRUE`, `lty=1:2`, `cex=0.7` (see help of the function for more details).

- `ginimap`: Lorentz curve from `name.var` and calculates the Gini Index associated to `name.var`.
- `histomap`: histogram of a given variable `name.var`. Specific option are `nbcol` for the number of bars and `type` for the values to print on y-axis (`count`, `percent` or `density`)
- `variocloudmap`: semi-variocloud (directional or omnidirectional) and a map. Specific options are `bin` which indicates the x-axis where the variocloud will be evaluated and `quantiles` for drawing a conditionnal quantile curve.

3.2 Functions with several variables of interest

- `clustermap`: classification of the sites from the variables called in `names.var` and computes a bar plot of the clusters calculated. Specific options are `clustnum` which gives the number of cluster, `method`, which gives the method to use, `type`, `center`, `scale` which gives indication on the method (see `help(clustermap)`) and `names.arg`.
- `dbledensitymap`: two kernel density estimates from 2 variables. Specific option is `kernel` for the choice of kernel.
- `dblehistomap`: two histograms of the given variables `names.var[1]` and `names.var[2]`. Specific option are `nbcol` and `type`.
- `histobarmap`: bar plot (vertical bar) of the given variable `names.var[1]` and histogram of the given variable `names.var[2]`. Specific options are `type` and `names.arg`.
- `pcamap`: plots summarizing a generalized Principal Component Analysis (PCA), made with `genpca` (wrap function). It draws the scatterplot of the individuals projected on a chosen principal component plane (with their percentage of inertia), together with the scatterplot of the variables projected into the same plane with the quality of representation in order to interpret the principal component axes. Specific options are `direct`, `weight`, `metric`, `center`, `reduce` and `qualproj` (see `help(pcamap)`).
- `plot3dmap`: 3d-plot of three given variables *names.var*. Specific options are `box` for drawing a cube and `zlab`. It depends on package **rgl**.
- `polyboxplotmap`: parallel Boxplots of a numerical variable by levels of a factor. Specific options are `varwidth` and `names.arg`
- `scattermap`: scatterplot of the given variables indicated in `names.var`.

3.3 Function with variable(s) of interest and a spatial weight matrix

- `misolationmap`: scatterplot with the pairwise Mahalanobis distances calculated using variables `names.var` between the observations and their neighbors on the y-axis and the "degree of isolation" of the observations on the x-axis. It depends on **mvoutlier** and **robustbase** packages. Specific options are `propneighb` and `chisqqu`

- `moranplotmap`: moran plot, on the x-axis, is represented $x - \bar{x}$ and on the y-axis $W(x - \bar{x})$, where W is the spatial weight matrix. It also calcultes Moran's I statistic (see `nonnormoran`) and give a p-value associated to the autocorrelation test (gaussian version and permutation version). Specific options are `flower`, `locmoran` and `names.arg`.

- `mvariocloudmap`: scatterplot of pairwise Mahalanobis distances and spatial distances with a map. It is a multivariate version of the variocloud. The number of couples of sites plotted can be reduced by considering couples above a quantile regression curve. It depends on **mvoutlier** and **robustbase** packages. Specific option is `quantiles`.

- `neighbourmap`: scatterplot of the values of the variable at neighbouring sites for a neighbourhood structure given by a binary weight matrix W.

At these functions, we could add `barnbmap` and `histnbmap` which analyse the spatial neigborhood structure.

3.4 Other dependencies

The quantile spline regression drawn on the scatterplot with option `quantiles` comes from function `qsreg` included in **fields** package. **spalncs** package is called for the use of the `inout` function.

4 A example of Spatial econometric function

We present he the example proposed by the `neighbourmap` function and by using the same data set `immob`.

4.1 Construction of a spatial weight matrix

It exists several functions in **spdep** package which build spatial weight matrix. These functions create a `nb` object which corresponds to the class used in the **GeoXp** functions. For example, the `tri2nb` function build a spatial weight matrix based on triangulation Delaunay:

```
> W.nb <- tri2nb(cbind(immob$longitude, immob$latitude))
> class(W.nb)
[1] "nb"
```

In **GeoXp**, the function `makeneighborsw` can build a spatial weight matrix by using both method of the nearest neighbors and the threshold distance. However, the result is included in a `matrix` object and the user will have to convert this object into a `nb` object by using the function `mat2listw`, like this:

```
> W2.matrix <- makeneighborsw(cbind(immob$longitude, immob$latitude),
+     method = "both", m = 5, d = 175000)
> W2.nb <- mat2listw(W2.matrix)$neighbours
> class(W2.nb)

[1] "nb"
```

Notice that the functions `histnbmap` and `barnbmap` included in **GeoXp** could make the interactive analysis of the neighborhood structure given by a `nb` object.

4.2 Example of use of a spatial econometric function

In the following example, we consider the variable "average price of sell of house by square meter" and use the `neighbourmap`. We indicate as third element, the spatial weight matrix of class `nb`.

```
> neighbourmap(immob.spdf, "prix.vente", W.nb, identify = TRUE,
+     cex.lab = 0.5, carte = cont_midiP)
```

In these example, we have selected two cities on the map. The value of the city observed in the North corresponds on the scatterplot to the axis of the first column of points represented in red. The points represented in red in this column corresponds to the neighbours of the city selected on the map. The fact that the points are located above the line $y = x$ means that the city selected is a local "outlier" in the sense where the value taken is lower then its neighbours. For the second city selected in the South, that is the inverse.

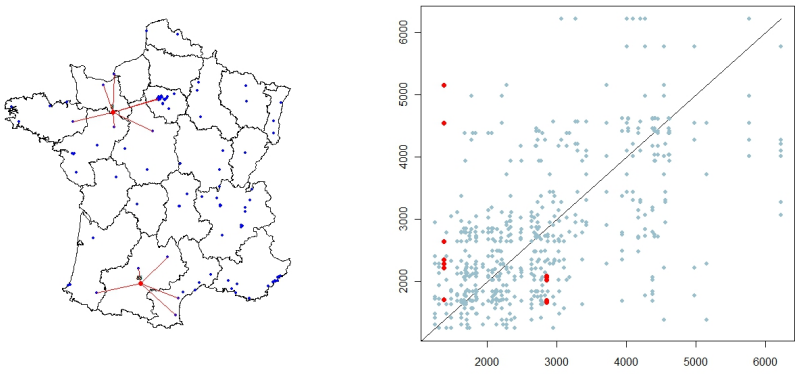

Figure 3: The use of options in histomap function

In this case, when user select the `Save result` button, it creates in the `last.select` object, a matrix with the couples of sites selected:

```
[1] "Results have been saved in last.select object"
```

```
> last.select
```

```
      [,1] [,2]
 [1,]    3   17
 [2,]    3   38
 [3,]    3   39
 [4,]    3   40
 [5,]    3   63
 [6,]    3   70
 [7,]    3   90
 [8,]   85   53
 [9,]   85   58
[10,]   85   65
[11,]   85   66
[12,]   85   73
```