Universitat Politècnica de València

School of Engineering in Geodesy, Cartography and Surveying

PERSONAL ASSIGNMENT 2

Desarrollo de aplicaciones SIG

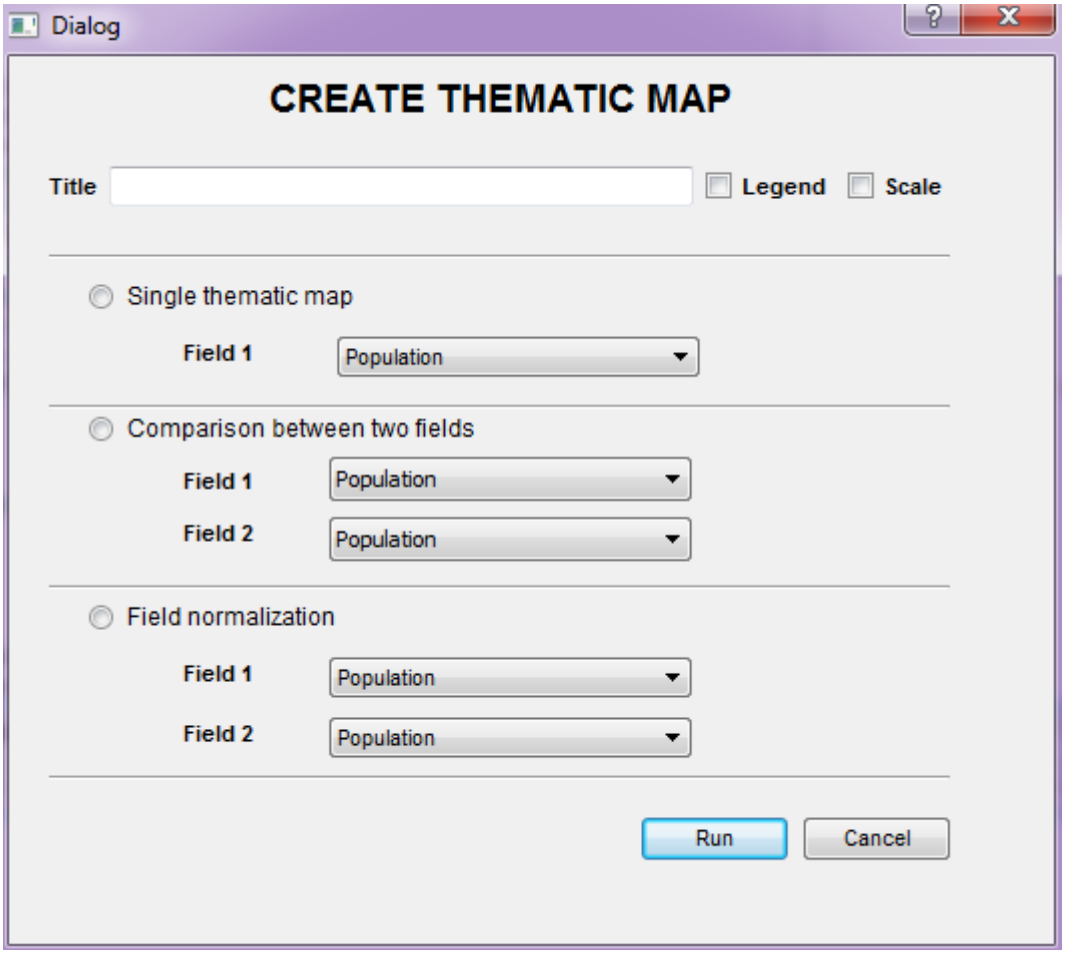
Eliška VLČKOVÁ

València 2017

INDEX

1 GOAL

The goal of this personal assignment is to make a standalone script in order to make several maps – defined by user. There are three choices: make single thematic map, make a map with comparison of two selected fields and the last one is map with normalized field. The user's input is also the „title" and the opportunity to make a scale or legend.



2 DESCRIPTION

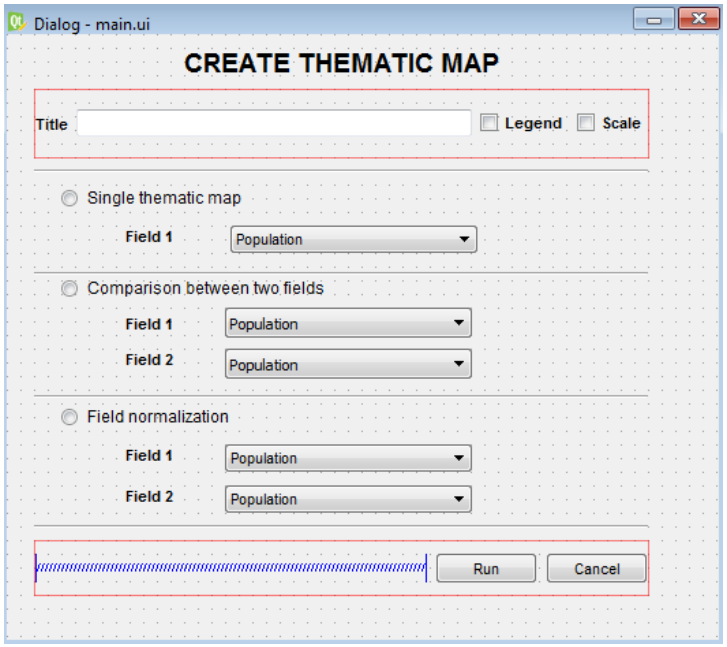
2.1 PyQt and Qt Designer

The PyQt installer comes with a GUI builder tool called **Qt Designer**. Using its simple drag and drop interface, a GUI interface can be quickly built without having to write the code. It is however, not an IDE such as Visual Studio. Hence, Qt Designer does not have the facility to debug and build the application.

Creation of a GUI interface using Qt Designer starts with choosing a top level window for the application.

It is really easy to drag and drop required widgets from the widget box on the left pane. I can also assign value to properties of widget laid on the form.

The designed form is saved as main.ui (see on the picture below). This ui file contains XML representation of widgets and their properties in the design.



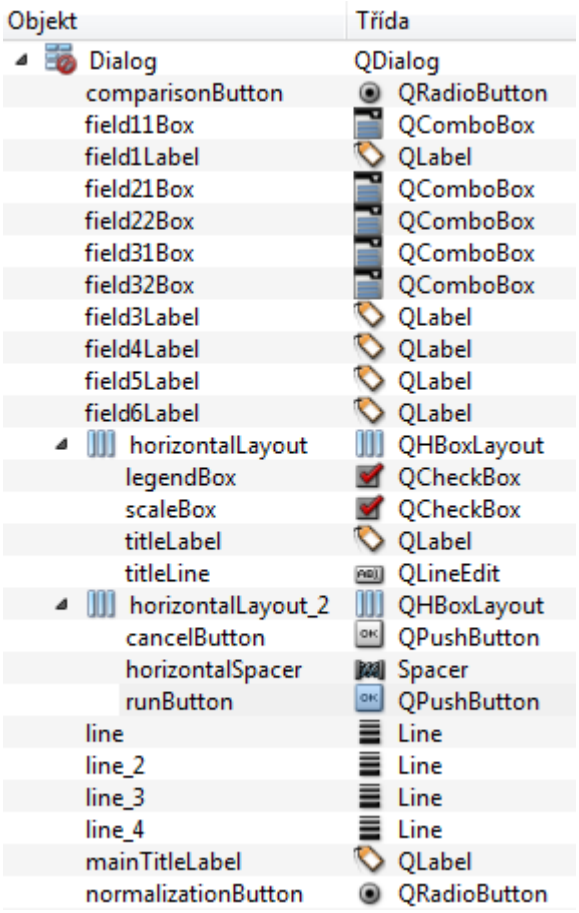
Unlike a console mode application, which is executed in a sequential manner, a GUI based application is event driven. Functions or methods are executed in response to user's actions like clicking on a button, selecting an item from a collection or a mouse click etc., called **events**.

Widgets used to build the GUI interface act as the source of such events. Each PyQt widget, which is derived from QObject class, is designed to emit '**signal**' in response to one or more events. The signal on its own does not perform any action. Instead, it is 'connected' to a '**slot**'. The slot can be any **callable Python function**. For example, „run" and „cancel" QPushButton call the run and cancel functions in the code. *self.runButton.clicked.connect(self.run)*

self.cancelButton.clicked.connect(self.cancel)

2.1.1 Objects

But each function demands different Object and class. In next picture can be seen, how were the objects used for this particular case.



QRadioButton and QCheckBox: QRadioButton and QCheckBox are both option buttons. That is, they can be switched on (checked) or off (unchecked). The classes differ in how the choices for the user are restricted. Check boxes define "many of many" choices, whereas radio buttons provide a "one of many" choice. In a group of radio buttons only one radio button at a time can be checked; if the user selects another button, the previously selected button is switched off.

QRadioButton: used for selection of map type – single map, comparing between two maps and normalizated map.

QComboBox: The QComboBox widget is a combined button and popup list. It provides a means of presenting a list of options to the user in a way that takes up the minimum amount of screen space. It is a selection widget that displays the current item, and can pop up a list of selectable items. A combobox may be editable, allowing the user to modify each item in the list.

QComboBoxes were used for user's field input. The fields were used the same as were available in feature class „provincias" – Population, Marriages, Births, Deaths, Unemployment Men and Women and the last added was area. This was done in order to do normalization of population field by area (to create density of population).

QPushButton: The push button, or command button, is perhaps the most commonly used widget in any graphical user interface. Push (click) a button to command the

computer to perform some action, or to answer a question. Typical buttons are OK, Apply, Cancel, Close, Yes, No and Help.

A command button is rectangular and typically displays a text label describing its action. A shortcut key can be specified by preceding the preferred character with an ampersand in the text. This button was used to perform „run“ and „cancel“ action.

QLineEdit: The QLineEdit widget is a one-line text editor. A line edit allows the user to enter and edit a single line of plain text with a useful collection of editing functions, including undo and redo, cut and paste, and drag and drop. In our case, we just needed to let user insert title of the map.

2.2 Python

2.2.1 Loading UI

Qt Designer uses XML .ui files to store designs and does not generate any code itself. Qt includes the uic utility that generates the C++ code that creates the user interface. Qt also includes the QUiLoader class that allows an application to load a .ui file and to create the corresponding user interface dynamically. The code is structured as a single class that is derived from the Python object type. The name of the class is the name of the toplevel object set in Designer with Ui_ prepended. (In the C++ version the class is defined in the Ui namespace.) We refer to this class as the *form class*.

The class contains a method called setupUi(). This takes a single argument which is the widget in which the user interface is created. The type of this argument (typically QDialog, QWidget or QMainWindow) is set in Designer.

Example of loading UI, which was used in our case can be seen on the picture below.

```
form = uic.loadUiType("main2.ui")[0]
class MyDialogClass(QtGui.QDialog, form):
    def __init__(self,parent=None):
        QtGui.QDialog.__init__(self,parent)
        self.setupUi(self)
        #run and calcel function
        self.runButton.clicked.connect(self.run)
        self.cancelButton.clicked.connect(self.cancel)

app = QtGui.QApplication(sys.argv)
myDialog = MyDialogClass(None)
myDialog.show()
app.exec_()
```

2.2.2 Accessing layers

Firstly, it needs to be located the path for the ArcMap Project. Because is my project saved in the same folder as the code is, it was used simly the order:

mxd = map.MapDocument(r"project.mxd"). Map is reference to library arcpy.mapping. After accessing to project, i needed to acces to dataframe (for single map there was only one dataframe and for comparing two tematic maps were used two dataframes). So the function ListDataFrames, what Returns a Python list of DataFrame objects that exist within a single map document (.mxd) was used. And finally, accesing layers. Because i

used 3 layers (the first and second layer were used for generating cartographic borders).So in that case the code was following: *layer = map.ListLayers(mxd,"",df)[2]*, because the layer i wanted to work with was in the third position.

2.3 Graphics

To acces the graphics in arcmap was used following code:

graphic_list = map.ListLayoutElements(mxd). This means, that all layout elements were assigned to graphic_list variable. And now, i can go through this in very simply loop proces: *for graphic in graphic_list:*

2.3.1 Title

The title of the map was composed from user's input (for what was used QLineEdit class in QDesigner).

```
if graphic.name == "mapTitle":
    graphic.text = str(self.titleLine.text())
    print "Title is",  graphic.text
```

The element name of Title in the ArcMap was mapTitle. If there will be found mapTitle, it will be filled with user input titleLine from graphical interface.

2.3.2 Scale

The scale was created in the scale 1 : 6 000 000 (in order to print for A4 paper or at A3 format for maps comparisment. Similar proces as with title was used for generating scale (and legend as well).

```
if graphic.name == "mapScale":
    if self.scaleBox.isChecked()==True:
        graphic.elementPositionX = 19.7365
        graphic.elementPositionY = 2.9956
    if self.scaleBox.isChecked()==False:
        graphic.elementPositionX = 1
        graphic.elementPositionY = -5
```

In scale can user decide, if he wants to generate it, or not. If yes, the prepared scale will appear on defined position. If no, the scale will appear as well, but out of the range of map layout, so will not be exported to final map. The scale was prepared before and for edition it was converted to graphics.

2.3.3 Legend

Firstly, in case to follow all cartographic rules, the legend was converted to graphics in the same proces as the scale. But in the end, the maps didn't generate the new legend with elements based on input values. So this is the reason, why Legend doesn't follow all cartographic rules.

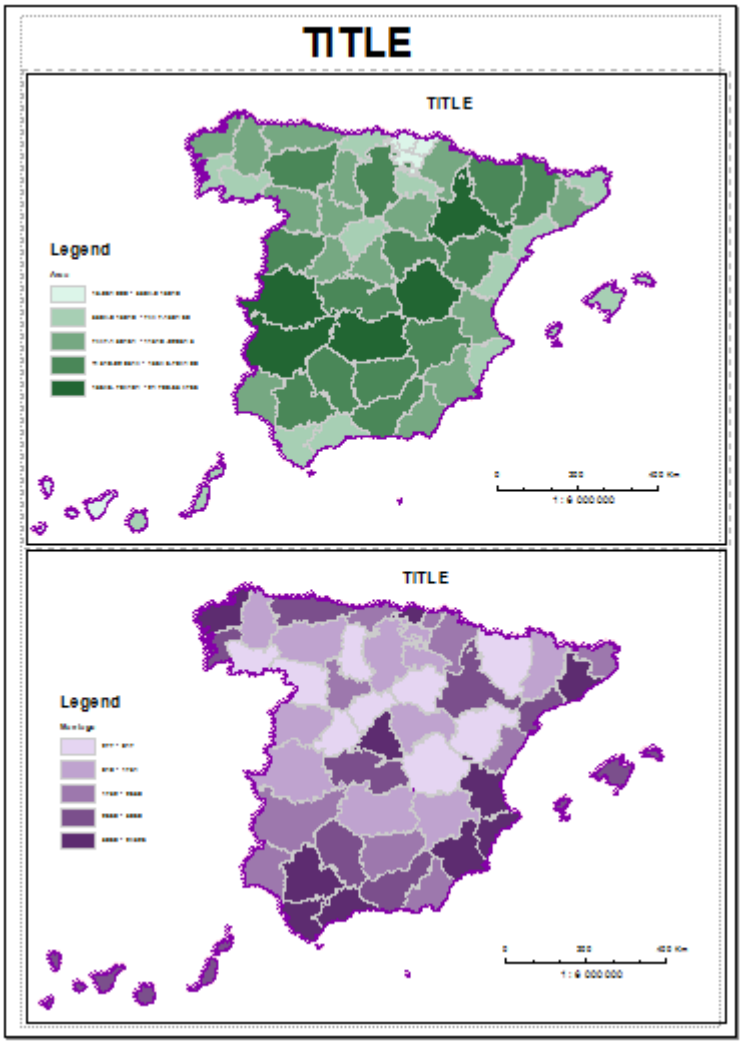
```
if graphic.name == "mapLegend":
    if self.legendBox.isChecked()==True:
        graphic.elementPositionX = 2.4086
        graphic.elementPositionY = 8.1774
    if self.legendBox.isChecked()==False:
        #pass
        graphic.elementPositionX = 1
        graphic.elementPositionY = -5
```

The code above showes, how was used the legend generation. It is very similar to scale generation.

2.4 Symbology

2.4.1 .lyr creating

Firstly, for all variables were created specific layer files. The layer was used once, but with specific symbology for the variables in order to create graduated colours. The classify function was established as quantiles. On the picture below, can be seen how was created the „template“ for next map generation.



2.4.2 Accessing to .lyr layers through python

Based of user's input (here is the example for creating single thematic map) is the layer saved to variable layerSymbology, so we can work with it in next step.

```
if str(self.field11Box.currentText())== "Population":
    layerSymbology = "layers//population.lyr"
if str(self.field11Box.currentText())== "Marriages":
    layerSymbology = "layers//marriages.lyr"
if str(self.field11Box.currentText())== "Births":
    layerSymbology = "layers//narozeni.lyr"
if str(self.field11Box.currentText())== "Deaths":
    layerSymbology = "layers//deaths.lyr"
if str(self.field11Box.currentText())== "Unemployment Men":
    layerSymbology = "layers//unpl_men.lyr"
if str(self.field11Box.currentText())== "Unemployment Women":
    layerSymbology = "layers//unpl_women.lyr"
if str(self.field11Box.currentText())== "Area (km2)":
    layerSymbology = "layers//area.lyr"
```

2.5 PDF creating

After setting the symbology on our specific layer, is this symbology applied on our layer. After this is this layer with our applied symbology generated to PDF.

```
arcpy.ApplySymbologyFromLayer_management (layer,layerSymbology)
map.ExportToPDF(mxd, r"results\single_map.pdf")
```

2.6 Maps

For generating **single thematic map** were used the code as on examples above. This generate map with graduate symbols only for one field. For creating two **compared maps** wese used the similar approach, only with the difference of creating two data frames. So in the code, it was need to get acces to both data frames and after for both layers from these data frames. In this case was created another project, where were the maps generated on A3 paper format.

```
#-------TITLES-------
if graphic.name == "mapMainTitle":
    graphic.text = str(self.titleLine.text())
    print graphic.text
if graphic.name == "map1Title":
    graphic.text = str(self.field21Box.currentText())
if graphic.name == "map2Title":
    graphic.text = str(self.field22Box.currentText())
```

The titles were changed a bit as well. The main title remains as from the another creating single thematic map. But based on user input field for comparing maps was generated the title for each of them. Analogous, the legend and the scale map was created twice and put on two position; and also applying symbology had to be done on two different layers in two different dataframes. The last one remains creating the **normalized map.** For this map was used the same ArcMap project as for the sigle thematic map. But a few functions had to be added. Firstly, it was user definition on field for normalization.

```
if str(self.field32Box.currentText())== "Population":
    fieldN = "Population"
if str(self.field32Box.currentText())== "Marriages":
    fieldN = "Marriage"
if str(self.field32Box.currentText())== "Births":
    fieldN = "Birth"
if str(self.field32Box.currentText())== "Deaths":
    fieldN = "Death"
if str(self.field32Box.currentText())== "Unemployment Men":
    fieldN = "UnplMen"
if str(self.field32Box.currentText())== "Unemployment Women":
    fieldN = "UnplWomen"
if str(self.field32Box.currentText())== "Area (km2)":
    fieldN = "Area"
```

Based on differend fields, the variable fieldN will be assigned with the field in attribute table. With this assigning, and throught the code in the picture bellow, can bet he normalization done very easily.

```
arcpy.ApplySymbologyFromLayer_management (layer, layerSymbology)
symb = layer.symbology
symb.normalization = fieldN
```

After a few problems with ArcGIS version and normalization, it had to be done in different way. The layers were created for each option, and called through two conditions.

```
# ///// MARRIAGES
            if str(self.field31Box.currentText())== "Marriages":
                #layerSymbology = "layers//population.lyr"
                if str(self.field32Box.currentText())== "Marriages":
                    layerSymbology = "layers/marriages.lyr"
                    print "can not choose this object in both fields. Created
                if str(self.field32Box.currentText())== "Births":
                    layerSymbology = "layers/normalization/marbirth.lyr"
                if str(self.field32Box.currentText())== "Deaths":
                    layerSymbology = "layers/normalization/mardeath.lyr"
                if str(self.field32Box.currentText())== "Unemployment Men":
                    layerSymbology = "layers/normalization/marunplm.lyr"
                if str(self.field32Box.currentText())== "Unemployment Women":
                    layerSymbology = "layers/normalization/marunplw.lyr"
                if str(self.field32Box.currentText())== "Area (km2)":
                    layerSymbology = "layers/normalization/mararea.lyr"
                if str(self.field32Box.currentText())== "Population":
                    layerSymbology = "layers/normalization/marpop.lyr"
```

3 CODE

```
# Author:      Vlckova Eliska
# Created:     26/05/2017
# Copyright:   (c) EFOX 2017
# Licence:     <any creative commons>
#-------------------------------------------------------------------


import arcpy, os, tempfile, sys
import arcpy.mapping as map
from PyQt4 import QtCore, QtGui, uic

form = uic.loadUiType("main2.ui")[0]
class MyDialogClass(QtGui.QDialog, form):
    def __init__(self,parent=None):
        QtGui.QDialog.__init__(self,parent)
        self.setupUi(self)

        #run and calcel function
        self.runButton.clicked.connect(self.run)
        self.cancelButton.clicked.connect(self.cancel)

    #function for RUN
    def run(self):
        #SINGLE THEMATIC
        if self.singleButton.isChecked()==True:
            mxd = map.MapDocument(r"project.mxd")
            df = map.ListDataFrames(mxd)[0]
            layer3 = map.ListLayers(mxd,"",df)[0]
            layer2 = map.ListLayers(mxd,"",df)[1]
            layer = map.ListLayers(mxd,"",df)[2]
            graphic_list = map.ListLayoutElements(mxd)
            for graphic in graphic_list:
                if graphic.name == "mapTitle":
                    graphic.text = str(self.titleLine.text())
                    print "Title is",  graphic.text

                if graphic.name == "mapScale":
                    if self.scaleBox.isChecked()==True:
                        graphic.elementPositionX = 19.7365
                        graphic.elementPositionY = 2.9956
                    if self.scaleBox.isChecked()==False:
                        graphic.elementPositionX = 1
                        graphic.elementPositionY = -5
                if graphic.name == "mapLegend":
                    if self.legendBox.isChecked()==True:
                        graphic.elementPositionX = 2.4086
                        graphic.elementPositionY = 8.1774
                    if self.legendBox.isChecked()==False:
                        #pass
                        graphic.elementPositionX = 1
                        graphic.elementPositionY = -5
```

```
    if str(self.field11Box.currentText())== "Population":
        layerSymbology = "layers//population.lyr"
    if str(self.field11Box.currentText())== "Marriages":
        layerSymbology = "layers//marriages.lyr"
    if str(self.field11Box.currentText())== "Births":
        layerSymbology = "layers//narozeni.lyr"
    if str(self.field11Box.currentText())== "Deaths":
        layerSymbology = "layers//deaths.lyr"
    if str(self.field11Box.currentText())== "Unemployment Men":
        layerSymbology = "layers//unpl_men.lyr"
    if str(self.field11Box.currentText())== "Unemployment Women":
        layerSymbology = "layers//unpl_women.lyr"
    if str(self.field11Box.currentText())== "Area (km2)":
        layerSymbology = "layers//area.lyr"

    arcpy.ApplySymbologyFromLayer_management (layer,layerSymbology)
    map.ExportToPDF(mxd, r"results\single_map.pdf")

#------COMPARING-------
if self.comparisonButton.isChecked()==True:
    mxd = map.MapDocument(r"project2.mxd")
    df1 = map.ListDataFrames(mxd)[0]
    layer13 = map.ListLayers(mxd,"",df1)[0]
    layer12 = map.ListLayers(mxd,"",df1)[1]
    layer11 = map.ListLayers(mxd,"",df1)[2]

    df2 = map.ListDataFrames(mxd)[1]
    layer23 = map.ListLayers(mxd,"",df2)[0]
    layer22 = map.ListLayers(mxd,"",df2)[1]
    layer21 = map.ListLayers(mxd,"",df2)[2]

    #GRAPHOCS ELEMENTS
    graphic_list = map.ListLayoutElements(mxd)
    for graphic in graphic_list:
        #-------TITLES-------
        if graphic.name == "mapMainTitle":
            graphic.text = str(self.titleLine.text())
            print graphic.text
        if graphic.name == "map1Title":
            graphic.text = str(self.field21Box.currentText())
        if graphic.name == "map2Title":
            graphic.text = str(self.field22Box.currentText())
        #-------legends------
        if graphic.name == "legend1Map":
            if self.legendBox.isChecked()==True:
                graphic.elementPositionX = 2
                graphic.elementPositionY = 10
            if self.legendBox.isChecked()==False:
                graphic.elementPositionX = -20
                graphic.elementPositionY = -20
        if graphic.name == "legend2Map":
            if self.legendBox.isChecked()==True:
                graphic.elementPositionX = 2
                graphic.elementPositionY = 24
            if self.legendBox.isChecked()==False:
                graphic.elementPositionX = -22
                graphic.elementPositionY = -20
```

```
    if graphic.name == "scale1Map":
        if self.scaleBox.isChecked()==True:
            graphic.elementPositionX = 20
            graphic.elementPositionY = 4
        if self.scaleBox.isChecked()==False:
            graphic.elementPositionX = 1
            graphic.elementPositionY = -5
    if graphic.name == "scale2Map":
        if self.scaleBox.isChecked()==True:
            graphic.elementPositionX = 20
            graphic.elementPositionY = 23
        if self.scaleBox.isChecked()==False:
            graphic.elementPositionX = -20
            graphic.elementPositionY = -20


if str(self.field21Box.currentText())== "Population":
    layerSymbology = "layers//population.lyr"
if str(self.field21Box.currentText())== "Marriages":
    layerSymbology = "layers//marriages.lyr"
if str(self.field21Box.currentText())== "Births":
    layerSymbology = "layers//narozeni.lyr"
if str(self.field21Box.currentText())== "Deaths":
    layerSymbology = "layers//deaths.lyr"
if str(self.field21Box.currentText())== "Unemployment Men":
    layerSymbology = "layers//unpl_men.lyr"
if str(self.field21Box.currentText())== "Unemployment Women":
    layerSymbology = "layers//unpl_women.lyr"
if str(self.field21Box.currentText())== "Area (km2)":
    layerSymbology = "layers//area.lyr"
arcpy.ApplySymbologyFromLayer_management (layer11,layerSymbology)

 if str(self.field22Box.currentText())== "Population":
     layerSymbology = "layers//population.lyr"
 if str(self.field22Box.currentText())== "Marriages":
     layerSymbology = "layers//marriages.lyr"
 if str(self.field22Box.currentText())== "Births":
     layerSymbology = "layers//narozeni.lyr"
 if str(self.field22Box.currentText())== "Deaths":
     layerSymbology = "layers//deaths.lyr"
 if str(self.field22Box.currentText())== "Unemployment Men":
     layerSymbology = "layers//unpl_men.lyr"
 if str(self.field22Box.currentText())== "Unemployment Women":
     layerSymbology = "layers//unpl_women.lyr"
 if str(self.field22Box.currentText())== "Area (km2)":
     layerSymbology = "layers//area.lyr"
 arcpy.ApplySymbologyFromLayer_management (layer21,layerSymbology)

 map.ExportToPDF(mxd, r"results//comparing.pdf")
```

```
#----NORMALIZED
elif self.normalizationButton.isChecked()==True:
    path_mxd = r"project3.mxd"
    mxd = map.MapDocument(path_mxd)
    dataframe = map.ListDataFrames(mxd)[0]
    layer = map.ListLayers(mxd,"",dataframe)[0]
    #-----GRAPHICS
    graphic_list = map.ListLayoutElements(mxd)
    for graphic in graphic_list:
        if graphic.name == "mapTitle":
            graphic.text = str(self.titleLine.text())
            print "Title: ", graphic.text
        if graphic.name == "mapScale":
            if self.scaleBox.isChecked()==True:
                graphic.elementPositionX = 19.7365
                graphic.elementPositionY = 2.9956
            if self.scaleBox.isChecked()==False:
                graphic.elementPositionX = 1
                graphic.elementPositionY = -5
        if graphic.name == "mapLegend":
            if self.legendBox.isChecked()==True:
                graphic.elementPositionX = 2.4086
                graphic.elementPositionY = 8.1774
            if self.legendBox.isChecked()==False:
                graphic.elementPositionX = 1
                graphic.elementPositionY = -5

# /////POPULATION
            if str(self.field31Box.currentText())== "Population":
                #layerSymbology = "layers//population.lyr"
                if str(self.field32Box.currentText())== "Marriages":
                    layerSymbology = "layers/normalization/popmar.lyr"
                if str(self.field32Box.currentText())== "Births":
                    layerSymbology = "layers/normalization/popbirth.lyr"
                if str(self.field32Box.currentText())== "Deaths":
                    layerSymbology = "layers/normalization/popmar.lyr"
                if str(self.field32Box.currentText())== "Unemployment Men":
                    layerSymbology = "layers/normalization/popunplm.lyr"
                if str(self.field32Box.currentText())== "Unemployment Women":
                    layerSymbology = "layers/normalization/popunplw.lyr"
                if str(self.field32Box.currentText())== "Area (km2)":
                    layerSymbology = "layers/normalization/poparea.lyr"
                if str(self.field32Box.currentText())== "Population":
                    layerSymbology = "layers/population.lyr"
                    print "you created single map population"
```

```
 # ///// AREA
             if str(self.field31Box.currentText())== "Area (km2)":
                 #layerSymbology = "layers//population.lyr"
                 if str(self.field32Box.currentText())== "Marriages":
                     layerSymbology = "layers/area.lyr"
                     print "does not work. Try to use Area as normalization field in opossite way. Created single area map."
                 if str(self.field32Box.currentText())== "Births":
                     layerSymbology = "layers/area.lyr"
                     print "does not work. Try to use Area as normalization field in opossite way.Created single area map."
                 if str(self.field32Box.currentText())== "Deaths":
                     layerSymbology = "layers/area.lyr"
                     print "does not work. Try to use Area as normalization field in opossite way.Created single area map."
                 if str(self.field32Box.currentText())== "Unemployment Men":
                     layerSymbology = "layers/area.lyr"
                     print "does not work. Try to use Area as normalization field in opossite way.Created single area map."
                 if str(self.field32Box.currentText())== "Unemployment Women":
                     layerSymbology = "layers/area.lyr"
                     print "does not work. Try to use Area as normalization field in opossite way.Created single area map."
                 if str(self.field32Box.currentText())== "Area (km2)":
                     layerSymbology = "layers/area.lyr"
                     print "does not work. Try to use Area as normalization field in opossite way.Created single area map."
                     layerSymbology = "layers/area.lyr"
                 if str(self.field32Box.currentText())== "Population":
                     layerSymbology = "layers/area.lyr"
                     print "does not work. Try to use Area as normalization field in opossite way.Created single area map."
# ///// MARRIAGES
             if str(self.field31Box.currentText())== "Marriages":
                 #layerSymbology = "layers//population.lyr"
                 if str(self.field32Box.currentText())== "Marriages":
                     layerSymbology = "layers/marriages.lyr"
                     print "can not choose this object in both fields. Created single map of marriages"
                 if str(self.field32Box.currentText())== "Births":
                     layerSymbology = "layers/normalization/marbirth.lyr"
                 if str(self.field32Box.currentText())== "Deaths":
                     layerSymbology = "layers/normalization/mardeath.lyr"
                 if str(self.field32Box.currentText())== "Unemployment Men":
                     layerSymbology = "layers/normalization/marunplm.lyr"
                 if str(self.field32Box.currentText())== "Unemployment Women":
                     layerSymbology = "layers/normalization/marunplw.lyr"
                 if str(self.field32Box.currentText())== "Area (km2)":
                     layerSymbology = "layers/normalization/mararea.lyr"
                 if str(self.field32Box.currentText())== "Population":
                     layerSymbology = "layers/normalization/marpop.lyr"
# ///// DEATHS
             if str(self.field31Box.currentText())== "Deaths":
                 #layerSymbology = "layers//population.lyr"
                 if str(self.field32Box.currentText())== "Marriages":
                     layerSymbology = "layers/normalization/deathmar.lyr"
                 if str(self.field32Box.currentText())== "Births":
                     layerSymbology = "layers/normalization/deathbirth.lyr"
                 if str(self.field32Box.currentText())== "Deaths":
                     layerSymbology = "layers/death.lyr"
                     print "can not choose this object in both fields. Created single map of deaths"
                 if str(self.field32Box.currentText())== "Unemployment Men":
                     layerSymbology = "layers/normalization/deathunplm.lyr"
                 if str(self.field32Box.currentText())== "Unemployment Women":
                     layerSymbology = "layers/normalization/deathunplw.lyr"
                 if str(self.field32Box.currentText())== "Area (km2)":
                     layerSymbology = "layers/normalization/deatharea.lyr"
                 if str(self.field32Box.currentText())== "Population":
                     layerSymbology = "layers/normalization/deathpop.lyr"
```

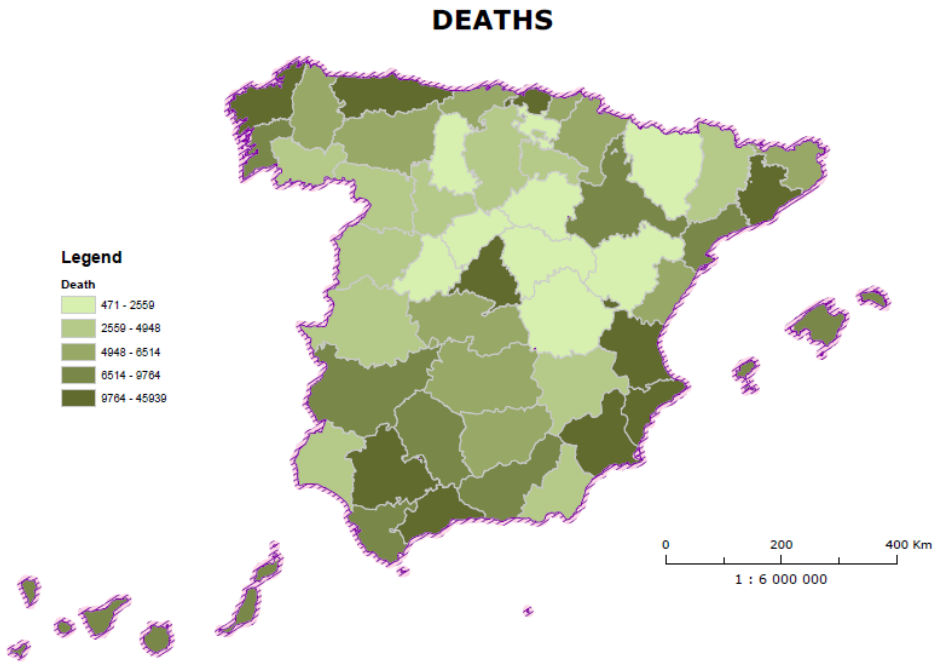
```
# ///// BIRTHS
            if str(self.field31Box.currentText())== "Births":
                #layerSymbology = "layers//population.lyr"
                if str(self.field32Box.currentText())== "Marriages":
                    layerSymbology = "layers/normalization/birmar.lyr"
                if str(self.field32Box.currentText())== "Births":
                    layerSymbology = "layers/narozeni.lyr"
                    print "can not choose this object in both fields. Created single map of births"
                if str(self.field32Box.currentText())== "Deaths":
                    layerSymbology = "layers/normalization/birdeath.lyr"
                if str(self.field32Box.currentText())== "Unemployment Men":
                    layerSymbology = "layers/normalization/birunplm.lyr"
                if str(self.field32Box.currentText())== "Unemployment Women":
                    layerSymbology = "layers/normalization/birunplw.lyr"
                if str(self.field32Box.currentText())== "Area (km2)":
                    layerSymbology = "layers/normalization/birarea.lyr"
                if str(self.field32Box.currentText())== "Population":
                    layerSymbology = "layers/normalization/birpop.lyr"
# ///// UNEMPLOYMNENT MEN
            if str(self.field31Box.currentText())== "Unemployment Men":
                #layerSymbology = "layers//population.lyr"
                if str(self.field32Box.currentText())== "Marriages":
                    layerSymbology = "layers/normalization/unplmmar.lyr"
                if str(self.field32Box.currentText())== "Births":
                    layerSymbology = "layers/normalization/unplmbir.lyr"
                if str(self.field32Box.currentText())== "Deaths":
                    layerSymbology = "layers/normalization/unplmdeath.lyr"
                if str(self.field32Box.currentText())== "Unemployment Men":
                    layerSymbology = "layers/unpl_men.lyr"
                    print "can not choose this object in both fields. Created single map of man's unemnployment"
                if str(self.field32Box.currentText())== "Unemployment Women":
                    layerSymbology = "layers/normalization/unplmunplw.lyr"
                if str(self.field32Box.currentText())== "Area (km2)":
                    layerSymbology = "layers/normalization/unplmarea.lyr"
                if str(self.field32Box.currentText())== "Population":
                    layerSymbology = "layers/normalization/unplmpop.lyr"
# ///// UNEMPLOYMNENT woMEN
            if str(self.field31Box.currentText())== "Unemployment Women":
                #layerSymbology = "layers//population.lyr"
                if str(self.field32Box.currentText())== "Marriages":
                    layerSymbology = "layers/normalization/unplwmar.lyr"
                if str(self.field32Box.currentText())== "Births":
                    layerSymbology = "layers/normalization/unplwbir.lyr"
                if str(self.field32Box.currentText())== "Deaths":
                    layerSymbology = "layers/normalization/unplwdeath.lyr"
                if str(self.field32Box.currentText())== "Unemployment Men":
                    layerSymbology = "layers/normalization/unplwunplm.lyr"
                if str(self.field32Box.currentText())== "Unemployment Women":
                    layerSymbology = "layers/unpl_women.lyr"
                    print "can not choose this object in both fields. Created single map of women's unemployment"
                if str(self.field32Box.currentText())== "Area (km2)":
                    layerSymbology = "layers/normalization/unplwarea.lyr"
                if str(self.field32Box.currentText())== "Population":
                    layerSymbology = "layers/normalization/unplwpop.lyr"

            arcpy.ApplySymbologyFromLayer_management (layer, layerSymbology)
            map.ExportToPDF(mxd, r"results//normalization.pdf")

    # function for CANCEL
    def cancel(self):
        self.close()

app = QtGui.QApplication(sys.argv)
myDialog = MyDialogClass(None)
myDialog.show()
app.exec_()
```

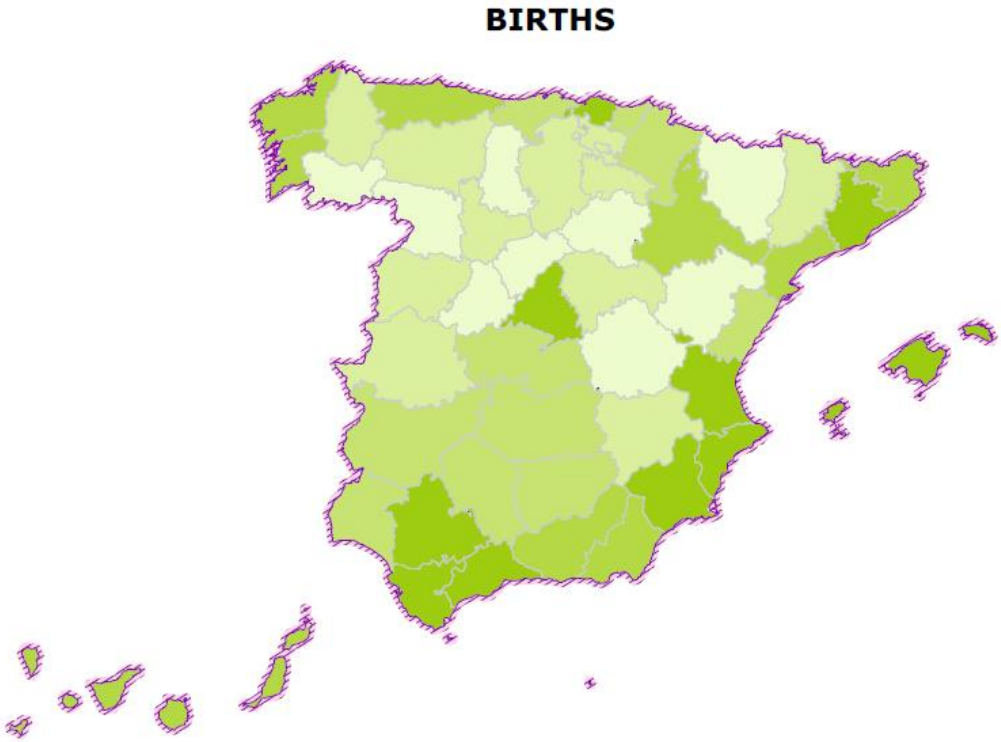
4 RESULTS

The standalone tool was tested several times. The pdf examples are saved in the results folder.

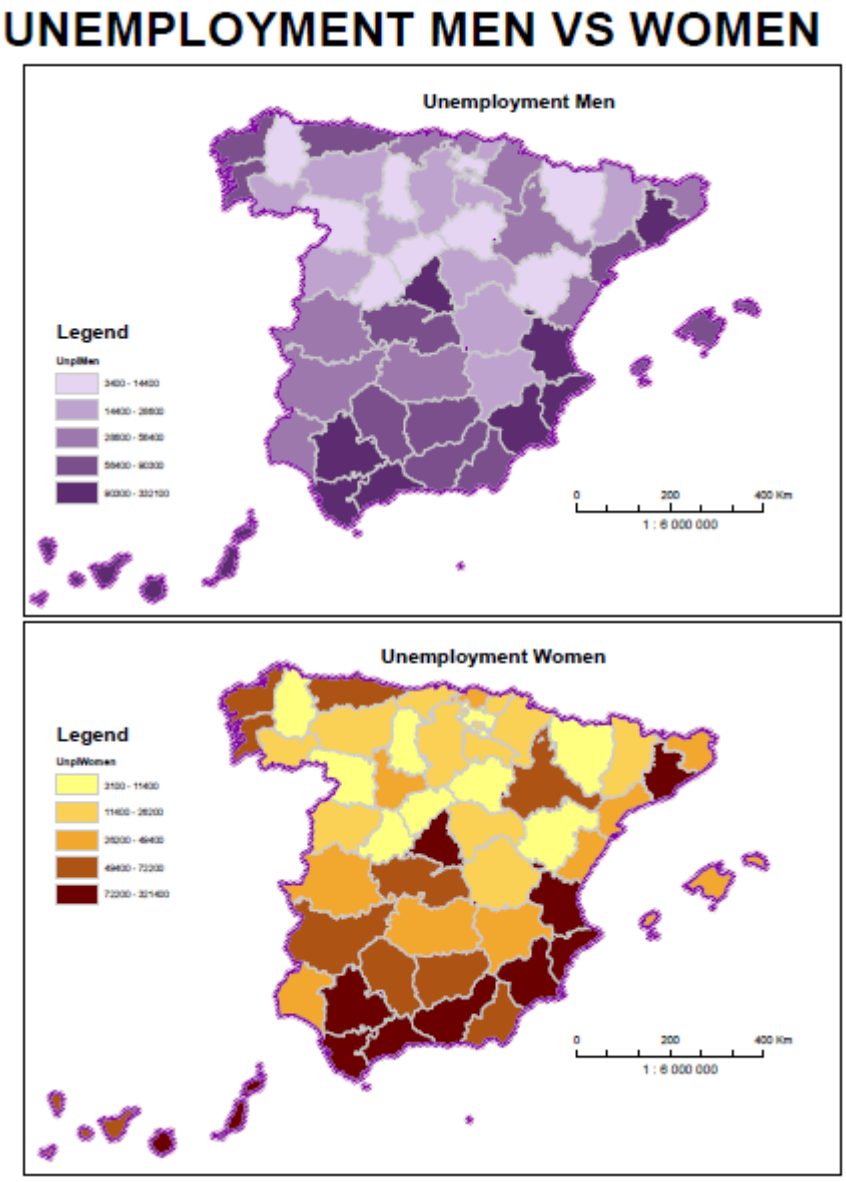
4.1 Single thematic map

The examples contains both version – with legend and scale, and without the legend and scale.

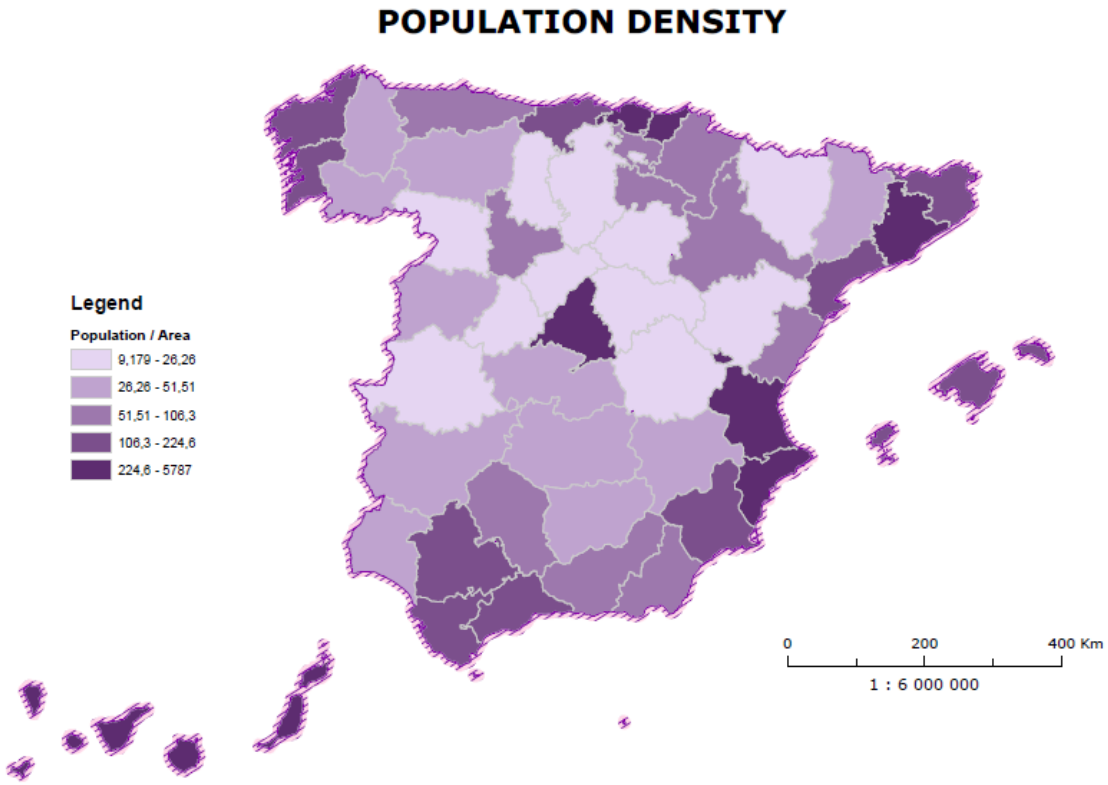




4.2 Comparison between two fields

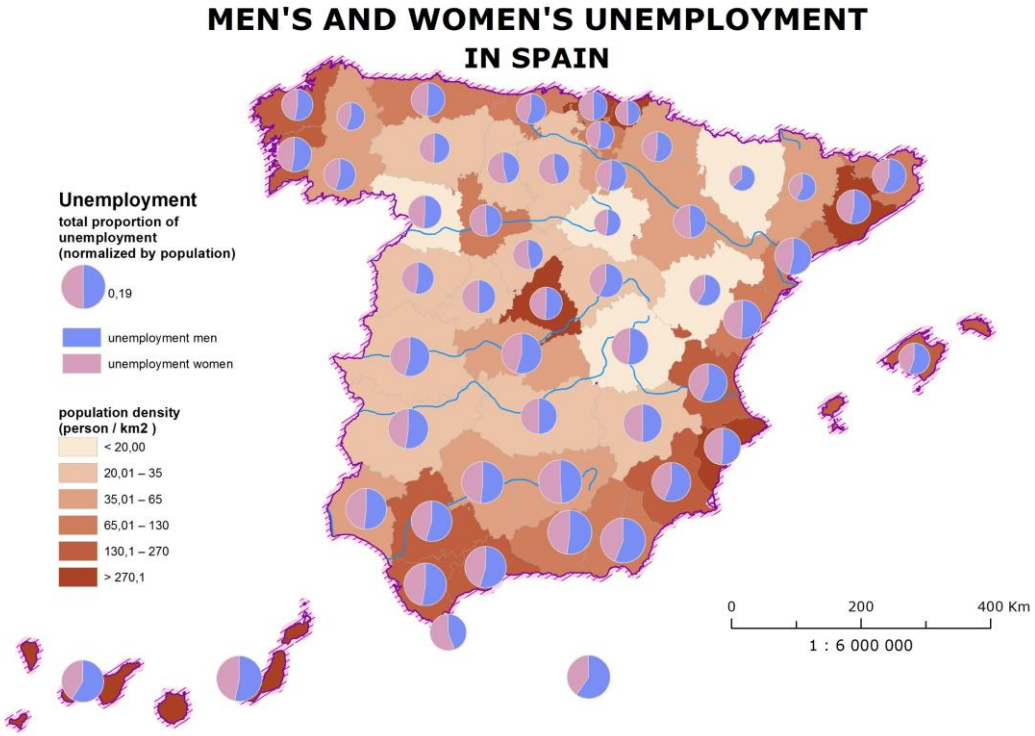


4.3 Normalization



5 CONCLUSION

Honestly, I still don't think that the standalone tool we created would be useful. For me is map generating still part of manuál proces, when is just necessary to repair the final map and follow the cartographic rules. It could be useful in case of data actualization. But firstly common error in choropleths is the use of raw data values to represent magnitude rather than normalized values to produce a map of densities. This is problematic because the eye naturally integrates over areas of the same color, giving undue prominence to larger polygons of moderate magnitude and minimizing the significance of smaller polygons with high magnitudes. In that case, instead of using choropleth map for „single thematic map", i would use diagram map. (the same for comparing two maps).



And moreover, i had to create all layers files before user's input and tool starting, so i didn't save any time at all. But in some cases (for normalization), it can be used – but the user should have any thoughs about the basic cartographic rules and to follow them. For example, that make the legend and the scale is basic and has to be on every map.

6 BIBLIOGRAPHY

- DIENER, Michael. *Python Geospatial Analysis Cookbook*. Packt Publishing Ltd. UK, 2015. ISBN 978-1-78355-507-9.

- *GDAL: gdaldem* [online]. Available in: http://www.gdal.org/gdaldem.html

- http://pythonforengineers.com/your-first-gui-app-with-python-and-pyqt/

- https://pythonprogramming.net

- https://www.cs.usfca.edu/~afedosov/qttut/

- https://www.safaribooksonline.com/blog/2014/01/22/create-basic-gui-using-pyqt/

- http://doc.qt.io

- http://www.learningpython.com/category/python/pyqt/#Introduction

- https://s3.amazonaws.com/webapps.esri.com/esri-proceedings/devsummit14/papers/dev-053.pdf

- http://pyqt.sourceforge.net/Docs/PyQt4/designer.html