GW framework

Lex Comber & Paul Harris

July 2016

Geographically Weighted Approaches

The previous exercise outlined how statistical approaches (logistic regression / Generalized Linear Models) could be used to generate probabilities of the User, Producer and Overall accuracy. These models are a-spatial because they consider all of the data in the study area, and they are referred to as *Global* models.

It is now possible to consider the geographically weighted extensions to these models (Brunsdon et al. 1996). For regression modelling, they are similar in form to ordinary regression, but geographically weighted approaches use a moving window or kernel under which local regressions are computed at locations throughout the study area.

To create GW models equations 2-4 in the previous exercise they are extended in the following way:

$$P(Ao = 1) = logit(b_{0(u_i,v_i)})(eqn1)$$

$$P(y_i = 1) = logit(b_{0(u_i,v_i)} + b_1 x_{1(u_i,v_i)}(eqn2)$$
$$P(x_i = 1) = logit(b_{0(u_i,v_i)} + b_1 y_{1(u_i,v_i)}(eqn3)$$

where $P(Ao = 1)$ is the overall accuracy probability at location i, x_1 and y_1 are the classified and reference data, respectively, and (u_i, v_i) is a vector of two-dimensional coordinates describing the location of i over which the coefficient estimates are assumed to vary. Thus, GW models are *local*: they are spatial dis-aggregations of *global* models and their outputs are location specific.

Critical to the operation of GW models is a moving window or kernel. The kernel moves through the study area (for example to cells in a predefined grid) and at each location computes a local model. It uses data under the kernel to construct a (local) model at that location with data weighted such that points further away from the kernel centre contribute less to the solution. Hence the *weighted* element in the GW framework. Thus, the kernel defines the data and weights that are used to calibrate the model at each location. The weight, w_i, associated with each location (u_i, v_i) is commonly a decreasing function of d_i, the distance from the centre of the kernel to (u_i, v_i):

A typical kernel function for example is the *bisquare* kernel. For a given bandwidth h, this is defined by

$$f(d) = \begin{cases} \left(1 - \left(\frac{d}{h}\right)^2\right)^2 & \text{if } d < h; \\ 0 & \text{otherwise.} \end{cases}$$

Here h may be defined as a fixed distance value, or in an adaptive distance way, for example to be the distance from the kth closest point in the `SpatialCrossTabs` object to $\mathbf{u}$. Gollini *et al* (2015) provide a description of common kernel shapes used in GW models. Generally larger values of h result in a greater degree of spatial smoothing - having a larger window around $\mathbf{u}$ in which cross tabulations have non-zero weighting.

Here, the bandwidth was selected to include the nearest 15% of the data points - see the code and detail of this below.

GW models in R

There are a number of packages that support GW analyses in R, including `spgwr`, `GWmodel`, `gwrr` and `McSpatial`. `GWmodel` is curated and supported by the original GWR team but `spgwr` is used here.

Gollini at al (2015) provide an overview of GW approaches and provide a thorough treatment demonstrating their use, including the steps of model selection, bandwidth / kernel size optimisation, handling collinearity etc. It also describes other GW models not based on regression, such as GW PCA.

The next sections deal with these stages in turn, focusing on GW generalized regression for the purpose of classification accuracy.

Set up

The code below loads the packages, the data and creates a `SpatialPointsDataFrame` variable for use as input to the GW generalized regression to determine overall accuracy. The packages can be called and the data can be loaded into R.

This can be done locally after you have set the working directory using the `setwd()` function. The example from my computer is below:

```
library(GISTools)
library(spgwr)
setwd("/Users/geoaco/Desktop/my_docs_mac/leeds_work/workshop/datacode")
roilib <- readShapePoly("lib_clip.shp")
data <- read.csv("all_data_new2.csv")
```

Alternatively the code below loads into your working directory directly from a `RData` file also on the site.

```
library(GISTools)
library(spgwr)
library(repmis)
source_data("http://github.com/lexcomber/LexTrainingR/blob/master/SpatialAccuracy.RData?raw=True")
```

```
## [1] "data"   "roilib"
```

And have a look at the what you have

```
ls()
```

```
## [1] "data"   "roilib"
```

```
head(data.frame(data))
```

```
##   PointID   East   North Urban_FS Vegetation_FS Woody_FS Grazing_FS
## 1       1 301847 3631819   0.2500         0.250    0.250       0.25
## 2       2 302491 3632155   0.0000         0.250    0.000       0.00
## 3       3 303834 3631818   0.0000         0.000    0.750       0.00
## 4       4 304480 3631008   0.2500         0.625    0.000       0.00
## 5       5 306691 3632967   0.6875         0.250    0.000       0.00
## 6       6 308175 3630784   0.0000         0.750    0.125       0.00
##   Bare_FS Boolean_FS Urban_RS Vegetation_RS Woody_RS Grazing_RS Bare_RS
## 1  0.0000          U    0.103         0.189    0.673      0.000    0.032
## 2  0.7500          B    0.256         0.036    0.387      0.000    0.321
## 3  0.2500          W    0.000         0.076    0.216      0.053    0.651
## 4  0.1250          V    0.112         0.372    0.215      0.185    0.110
## 5  0.0625          U    0.265         0.473    0.147      0.000    0.112
## 6  0.1250          V    0.000         0.365    0.312      0.143    0.175
##   Boolean_RS
## 1          V
## 2          B
## 3          W
## 4          V
## 5          V
## 6          V
```

Now create a variable indicating where the Field data / class is equal to the Classified data / class:

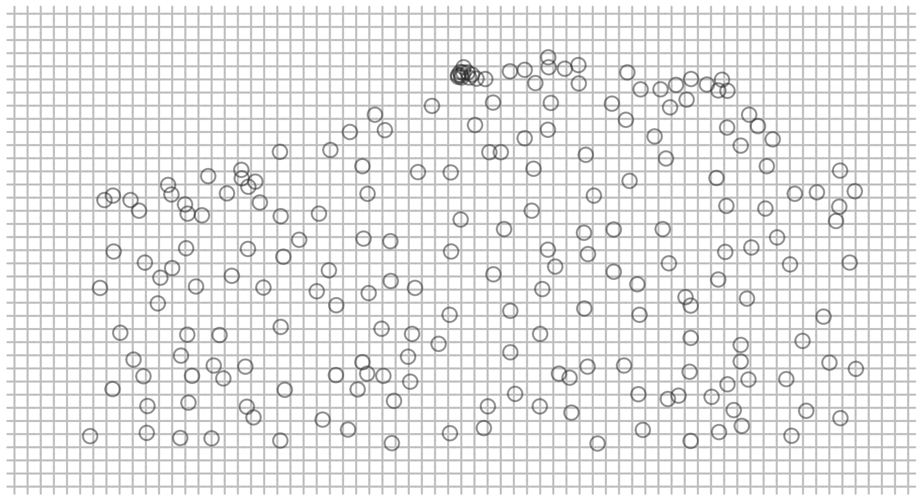
```
res <- vector(length = dim(data)[1])
for (i in 1: dim(data)[1]) {
    if (data$Boolean_RS[i] == data$Boolean_FS[i]) {
        res[i] <- 1
    }}
res.spdf <- SpatialPointsDataFrame(coords = data[,2:3], data = data.frame(res))
```

We also need to define the a logit function:

```
alogit <- function(x){exp(x)/(1+exp(x))}
```

In this case we want to calculate local models at each location on a 1km grid. This grid is created and mapped with the data in the code below:

```
grid <- SpatialPoints(expand.grid(x=seq(295000,363000,by=1000),
  y=seq(3610000,3646000,by=1000)))
plot(grid, cex = 0.7, col = "grey")
plot(res.spdf, add = T, pch = 1, col = "#25252580")
```



Bandwidth specification

It is important to select a relevant bandwidth or kernel size that correctly reflects the spatial process. Too small and the results will possibly appear too heterogeneous, too large and any local variation will be smoothed out; and the results possibly appear too homogeneous. There are a number of automatic and objective bandwidth selection functions in `GWmodel` and in `spgwr`. For `spgwr` the one for GW GLMs is `ggwr.sel`.

```r
# NB the ggwr.sel will return warnings for logistic regression - ignore them
bw <- ggwr.sel(res~1, data = res.spdf,adapt = T, family= binomial, verbose = F)
```

```r
bw
```

```
## [1] 0.9999367
```

In this case the value is close to 1 suggesting that **all** of the data should be used at each location! Harry will explain the reasons for this.

Harry: This actually alludes to a stationary process, but is not a given. The default kernel is a discontinuous Gaussian kernel. Specifying an adpative bandwidth of 1 will still reflect a broadly non-stationry process as weights will distance-decay right up to the furthest away observations. Ideally, the function should allow adaptive bandwidths above 1 as values of 1000000 say, will allow all weights to tend to 1, and in doing so, more accurately reflect a stationary process. The use of the box-car kernel function with its 0 or 1 weighting scheme is one way around this, as an adaptive bandwidth of 1 directly entails a stationary process. This function can be written for use in `spgwr`. The box-car kernel is available in `GWmodel` precisely for these reasons.

So for pragmatism in this workshop context, the bandwidth was pre-set to 0.15 to ensure that for each local model the nearest 30 of the data points were included:

```r
bw = 0.15
```

Running the GW model: overall Accuuracy

Then the GW model can be used to generate coefficient estimates at each location which can be used to generate overall accuracy probabilities:

```r
# NB the ggwr will return warnings for logistic regression - ignore them
gwr.mod <- ggwr(res~1, data = res.spdf, adapt = bw,
  fit.points = grid, family= binomial)
gwr.ov <- alogit(data.frame(gwr.mod$SDF)[,2])# Compute local P(Ao=1)
```

Mapping the outputs

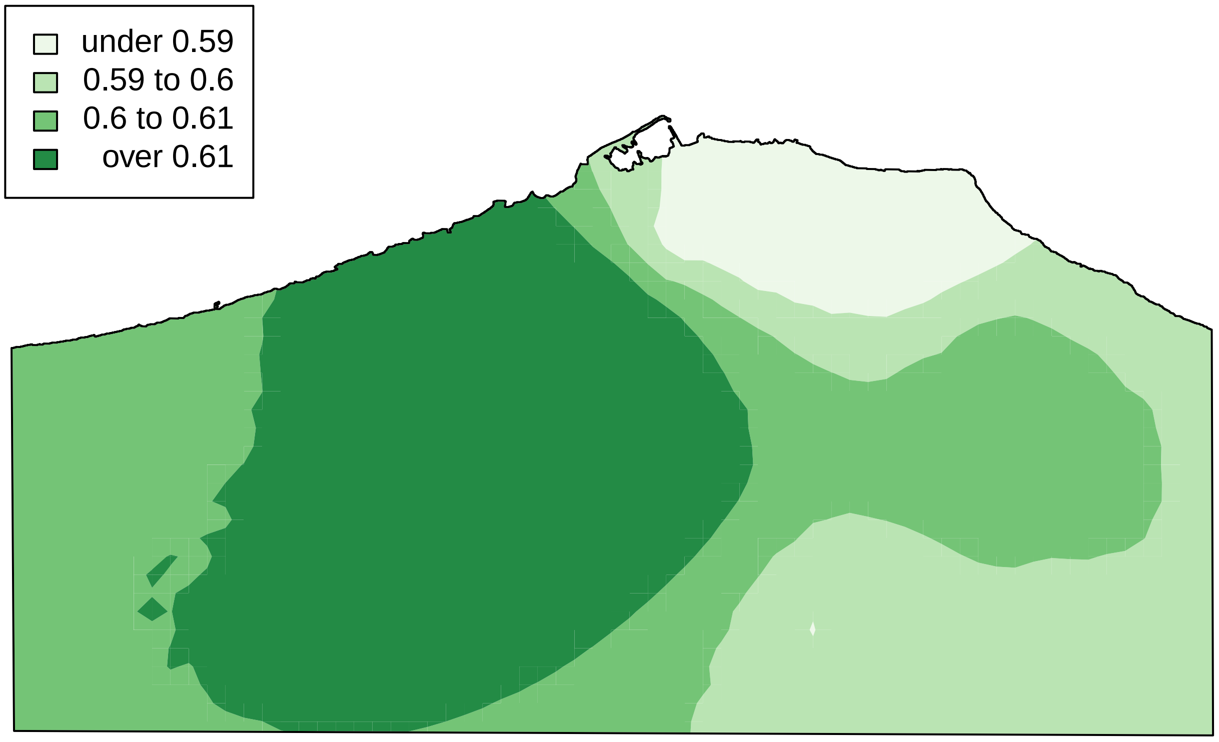
The results show only a small amount variation:

```r
summary(gwr.ov)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.5717  0.5962  0.6057  0.6053  0.6104  0.6446
```

But these can still be mapped

```
shades = auto.shading(gwr.ov, cols=brewer.pal(4,"Greens"))
gwr.res = SpatialPixelsDataFrame(gwr.mod$SDF, data.frame(gwr.ov))
par(mar = c(0,0,0,0))
level.plot(gwr.res,shades)
lib.masker = poly.outer(gwr.res, roilib, extend = 100)
add.masking(lib.masker)
plot(roilib, add = T)
choro.legend(297000, 3650000,shades)
```



Observe that it is hardly surprising that only broad spatial variations are evident, given the results of optimal bandwidth proceedure from before. That said, these subtle variations could still be important.

Summary

In this section you have been introduced to the basics of GW approaches and how they can be applied to validation data that are typically collected to generate *a*spatial or *global* accuracy measures. The GW framework constructs *local* models and allows spatially distributed measures of accuracy to be generated. In the next section you will produce User and producer measures of accuracy and map the results.

References

Brunsdon, C.F., Fotheringham, A.S. and Charlton M. (1996). Geographically Weighted Regression - A Method for Exploring Spatial Non-Stationarity, *Geographic Analysis*, 28: 281-298.

Gollini, I., Lu, B., Charlton, M., Brunsdon, C., and Harris, P. (2015). GWmodel: an R Package for Exploring Spatial Heterogeneity using Geographically Weighted Models. *Journal of Statistics*, 63(17): 1:50