

Assignment Two: The MAUP and Multilevel Modelling

Contents

1	Demonstrating the MAUP	1
1.1	Background	1
1.2	Demonstrating the MAUP	2
2	Multilevel Modelling	7
2.1	Model specification	7
2.2	Analysis	9
2.3	Interpretation	9
3	Bibliography	9
4	Appendix	9

1 Demonstrating the MAUP

1.1 Background

Areal units in zoning systems amalgamate into objects that constitute the basic units for the observation and analysis of spatial phenomena (Openshaw 2015). Yet, no gold standard for guiding the spatial aggregation process exists, with the validity of zonal objects subject to the arbitrary and modifiable decision-making of quantitative geographers. Problematically, the analysis of socioeconomic data involving areal units is encumbered by the modifiable areal unit problem (MAUP): “the sensitivity of analytical results to the definition of units for which data are collected.” According to the literature, the MAUP constrains the reliability of analyses for aggregated spatial data, as findings have shown varying results with the scale of aggregation and configuration of zoning systems (Avery and Clark 2015).

In practice, the MAUP is condensed into two issues of scale and zoning sensitivity which this paper will attempt to demonstrate in Section 1.2. The first issue, described as the *scale problem*, is the variation in findings when data for zonal units are progressively aggregated. This has been demonstrated empirically by Avery and Clark (2015) who found that whilst correlation coefficients did not increase monotonically with aggregation¹, a general increase in data aggregation corresponds to an increase in correlation coefficients.

The second issue, the *zoning problem*, pertains to the variation in findings when alternative combinations of zonal units are analysed with the scale or number of units held constant (Openshaw 2015). Zoning sensitivity in multivariate analysis has been demonstrated empirically in Fotheringham and Wong (2015) who simulated the aggregation of 871 block groups into 218 zones in 150 different iterations. They highlight the severity of the zoning problem by demonstrating the possibility of concluding with one iteration of zones no association between the percentage of blue-collar workers and mean family income, with another iteration finding a unit increase in blue-collar worker percentages as reducing mean family income by \$20,000. In all, ignoring scale and zoning sensitivity in model calibration can lead to inferential conclusions that a researcher’s areal data is applicable to the constituents who form the zones under study - the ecological fallacy problem (Openshaw 2015).

¹At higher levels of aggregation, there is smaller adjacency of zonal units meaning groupings are more heterogenous leading to lower correlation coefficients.

1.2 Demonstrating the MAUP

1.2.1 Data

To demonstrate the scaling and zoning sensitivities of the MAUP, we calculate the bivariate strength of association between two open data variables. For our first variable, *crime_count*, we submit a HTTP request using the POST verb to send a custom polygon for retrieving all street-level crimes occurring in 2012.

```
# download geojson
u <- "http://statistics.data.gov.uk/boundaries/E08000012.json"
# store in temporary directory
downloader::download(url = u, destfile = "/tmp/lpool.geojson")
lpool <- readOGR(dsn = "/tmp/lpool.geojson", layer = "OGRGeoJSON")
# access coords slot
lpool <- lpool@polygons[[1]]@Polygons[[1]]@coords
# build lat/lon + date string to send with postrequest
curl.string <- paste0('poly=',paste0(sprintf('%s,%s',lpool[,2], lpool[,1])
, collapse = ':'))

# build dates list for loop
dates = c("2012-01", "2012-02", "2012-03", "2012-04", "2012-05", "2012-06",
          "2012-07", "2012-08", "2012-09", "2012-10", "2012-11", "2012-12")

document <- lapply(dates, function(month) {
  # format acceptable packet for http request
  curl.string <- list(poly=c(curl.string), date=c(month))
  # post custom polygon to police api
  r <- httr::POST("https://data.police.uk/api/crimes-street/all-crime",
                  body = curl.string, encode="multipart", verbose())
  json <- content(r, "text", encoding = "ISO-8859-1")
  # return as data.frame
  jsonlite::fromJSON(txt=json)
})

# cast lat/lon columns to numeric data type
document$lat <- as.numeric(document$lat)
document$lon <- as.numeric(document$lon)

# convert data.frame to shapefile in rgdal
coordinates(document) <- ~lon+lat
proj4string(document) <- CRS("+init=epsg:4326") # WGS 84

# subsetting operations
document <- document[document$category == "violent-crime",]

# write once
writeOGR(document, "/Users/samcomber/Documents/spatial_analysis/shp/crimes",
          "crimes", driver = "ESRI Shapefile")

crimes <- readOGR(dsn = "/Users/samcomber/Documents/spatial_analysis/shp/crimes",
                  layer = "crimes")
crimes <- crimes[sample(nrow(crimes), 500), ]
```

Regarding our second variable, *tweet_count*, we aggregate geo-referenced tweets containing timestamps

relating to Twitter postings within the municipality of Liverpool for 2012².

```
# read twitter shapefile
tweets <- readOGR(dsn = "/Users/samcomber/Documents/spatial_analysis/shp/tweets",
                  layer = "tweets_mersey")

# sample 500 tweets
tweets <- tweets[sample(nrow(tweets), 500), ]
```

To demonstrate the MAUP, we aggregate a count of each crime and tweet into separate variables for each region in the shapefile, before computing correlations between the two vectors of values in the dataframe. We investigate the scaling problem using an iterative process that increments the number of regions in a hex-binned lattice from 10 to 100 (stepping each iteration by 10 regions). To demonstrate the zoning problem we constrain cardinality - i.e. the number of regions - to 30, and calculate correlation coefficients using 10 different permutations of 30 regions. To build random regions, we use the multiple aggregatons feature in IMAGE studio (???) which uses a depth-first search algorithm for a given contiguity matrix to create aggregated spatial regions. We then dissolve regions with the `unionSpatialPolygons` using the generated ZoneID for aggregations.

1.2.2 Scale problem

```
# directory loop
folders <- dir("/Users/samcomber/Documents/spatial_analysis/shp/img_stud/run1/")

# we have folders + file names and are in run1 directory
setwd("/Users/samcomber/Documents/spatial_analysis/shp/img_stud/run1/")

# create master dataframe to hold correlation coefficients
cors.scale.master <- data.frame()

# loop over aggregation folder - i.e. 10 regions, 20, 30
for (folder in folders) {
  setwd(paste0("/Users/samcomber/Documents/spatial_analysis/shp/img_stud/run1/", folder))

  # reread hex shapefile everytime we move folder
  hex <- readOGR(dsn = "/Users/samcomber/Documents/spatial_analysis/shp/hex", layer = "hex_1_clip")

  out <- data.frame()
  files <- list.files(pattern = "*.zon")

  # pick first iteration from each scale of aggregation
  img <- read.csv(files[1])

  # join regions to hex shapefile on area id
  hex@data$id <- as.numeric(hex@data$id)
  hex@data <- left_join(hex@data, img, by = c("id" = "AreaID"))
  hex@data <- hex@data[order(hex@data$id),]

  # dissolve by area id
  tes <- unionSpatialPolygons(hex, hex@data$ZoneID)
```

²This dataset was data mined by Guy Lansley from UCL, and processed by Dani Arribas-Bel.

```

# convert spatial polygons to spatialpolygonsdataframe
oldw <- getOption("warn")

# silence irrelevant warnings
options(warn = -1)
df <- data.frame(id = getSpPPolygonsIDSlots(tes))
row.names(df) <- getSpPPolygonsIDSlots(tes)
options(warn = oldw)

spdf <- SpatialPolygonsDataFrame(tes, data = df)

# points in polygon spatial join
pts.crimes <- point.in.poly(crimes, spdf)
pts.tweets <- point.in.poly(tweets, spdf)

# aggregate crimes/tweets data
pts.agg.crimes <- pts.crimes@data %>% group_by(id.1) %>% summarise(num = n()) %>% arrange(id.1)
pts.agg.tweets <- pts.tweets@data %>% group_by(id) %>% summarise(num = n()) %>% arrange(id)

# cast ids as factors for left_join
pts.agg.crimes$id.1 <- as.integer(pts.agg.crimes$id.1)
pts.agg.tweets$id <- as.integer(pts.agg.tweets$id)

# join summed crimes/tweets back to data.frame
hex@data <- left_join(hex@data, pts.agg.tweets, by = c("ZoneID" = "id"))
hex@data <- left_join(hex@data, pts.agg.crimes, by = c("ZoneID" = "id.1"))

# correlate unique zone id sums
uniques <- hex@data[!duplicated(hex@data$num.x), ]
cor.tc <- cor(uniques$num.x, uniques$num.y, use="complete.obs")

# rbind to master frame
cors.scale.master <- rbind(cors.scale.master, cor.tc)
}

cors.scale.master

```

Table 1: Correlations between crime and tweet counts.

<i>Number of regions</i>	<i>Correlation coeff.</i>
10	0.84
20	0.88
30	0.88
40	0.82
50	0.73
60	0.89
70	0.89
80	0.61
90	0.97
100	0.63

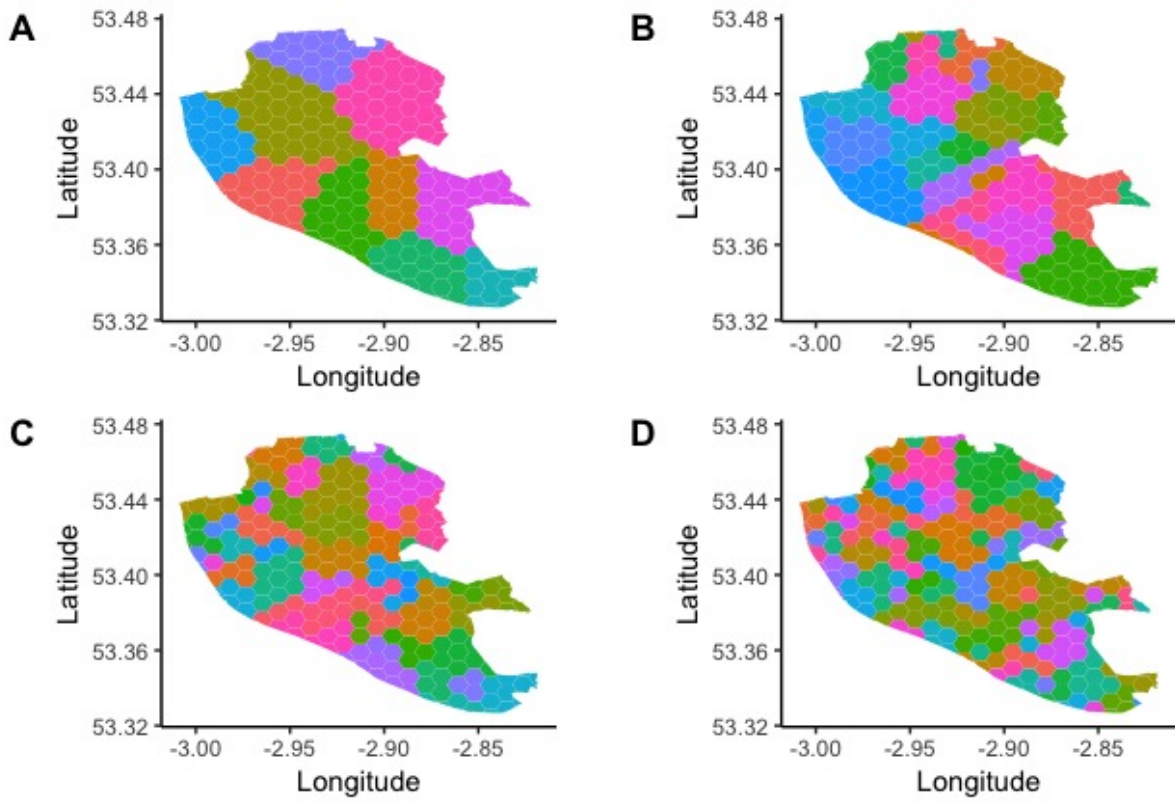


Figure 1: Scale aggregations. A shows 10 regions, B shows 30 regions, C shows 60 regions and D shows 100 regions.

1.2.3 Zoning problem

```
# we choose 20 regions to illustrate zoning problem
setwd("/Users/samcomber/Documents/spatial_analysis/shp/img_stud/run1/z070")

# get all 10 iterations of 20 regions
files.20 <- list.files(pattern = "*.zon")
cors.zone.master
# build master data.frame
cors.zone.master <- data.frame()
for (file in files.20) {
  hex <- readOGR(dsn = "/Users/samcomber/Documents/spatial_analysis/shp/hex", layer = "hex_1_clip")

  # read file to df
  zon <- read.csv(file)

  # join regions to hex shapefile on area id
  hex@data$id <- as.numeric(hex@data$id)
  hex@data <- left_join(hex@data, zon, by = c("id" = "AreaID"))
  hex@data <- hex@data[order(hex@data$id),]

  # dissolve by area id
  tes <- unionSpatialPolygons(hex, hex@data$ZoneID)

  # convert spatial polygons to spatialpolygonsdataframe
  oldw <- getOption("warn")

  # silence irrelevant warnings
  options(warn = -1)
  df <- data.frame(id = getSpPPolygonsIDSlots(tes))
  row.names(df) <- getSpPPolygonsIDSlots(tes)
  options(warn = oldw)

  spdf <- SpatialPolygonsDataFrame(tes, data = df)

  # points in polygon spatial join
  pts.crimes <- point.in.poly(crimes, spdf)
  pts.tweets <- point.in.poly(tweets, spdf)

  # aggregate crimes/tweets data
  pts.agg.crimes <- pts.crimes@data %>% group_by(id.1) %>% summarise(num = n()) %>% arrange(id.1)
  pts.agg.tweets <- pts.tweets@data %>% group_by(id) %>% summarise(num = n()) %>% arrange(id)

  # cast ids as factors for left_join
  pts.agg.crimes$id.1 <- as.integer(pts.agg.crimes$id.1)
  pts.agg.tweets$id <- as.integer(pts.agg.tweets$id)

  # join summed crimes/tweets back to data.frame
  hex@data <- left_join(hex@data, pts.agg.tweets, by = c("ZoneID" = "id"))
  hex@data <- left_join(hex@data, pts.agg.crimes, by = c("ZoneID" = "id.1"))

  # correlate unique zone id sums
  uniques <- hex@data[!duplicated(hex@data$num.x), ]
  cor.tc <- cor(uniques$num.x, uniques$num.y, use="complete.obs")
```

```

# rbind to master frame
cors.zone.master <- rbind(cors.zone.master, cor.tc)
}

```

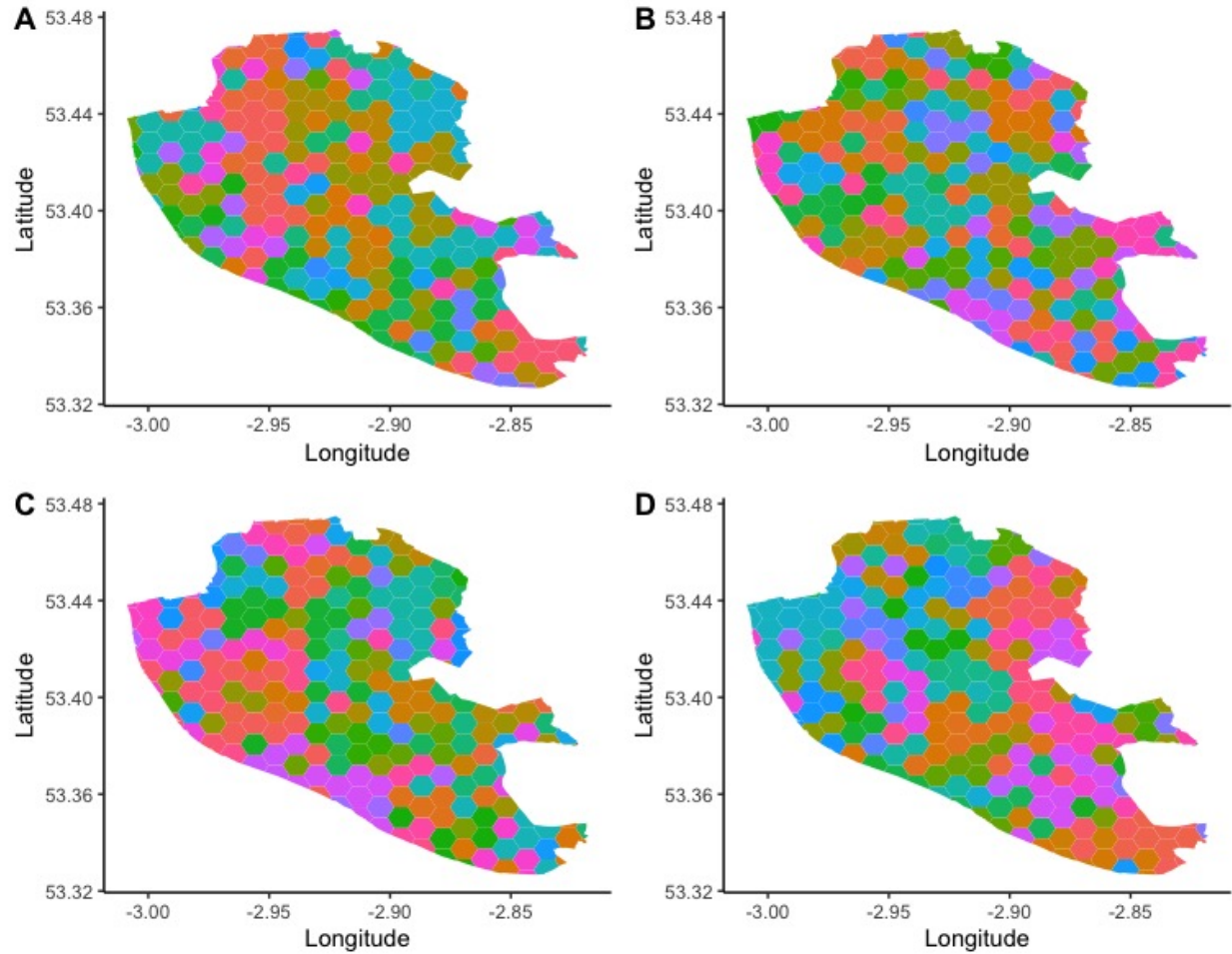


Figure 2: Zone aggregations. Four iterations displayed A-D.

From Table 2, we demonstrate the zoning problem: .

2 Multilevel Modelling

In this section, we implement a two-level multilevel model that accounts for spatial heterogeneity effects between the levels.

2.1 Model specification

```

# msoa and lsoa geographies were joined by location in QGIS
# as R lacks any robust library for polygon.in.polygon joins
lsoa.join <- readOGR(dsn =

```

Table 2: Correlation coefficients for 10 iterations of 70 regions.

<i>Iteration</i>	<i>Correlation coeff.</i>
1	0.89
2	0.66
3	0.61
4	0.89
5	0.42
6	0.96
7	0.75
8	0.92
9	0.82
10	0.84

```

        "/Users/samcomber/Documents/spatial_analysis/shp/spatial_join",
        layer = "lsoa_join", stringsAsFactors = FALSE)

# save location to tmp folder
temp <- "/tmp/lpool_pp.csv"

# download from source
download.file(paste0("http://landregistry.data.gov.uk/app/ppd/ppd_data.csv
?et%5B%5D=lrccommon%3Afreehold&et%5B%5D=lrccommon%3Aleasehold
&header=true&limit=all&max_date=31+December+2016
&min_date=1+January+2016&nb%5B%5D=true&nb%5B%5D=false
&ptype%5B%5D=lrccommon%3Adetached&ptype%5B%5D=lrccommon%3Asemi-detached
&ptype%5B%5D=lrccommon%3Aterraced&ptype%5B%5D=lrccommon%
3Aflat-maisonette&ptype%5B%5D=lrccommon%3AotherPropertyType
&tc%5B%5D=ppd%3AstandardPricePaidTransaction
&tc%5B%5D=ppd%3AadditionalPricePaidTransaction&town=",
"Liverpool"), temp)

# load LR data
lr <- read.csv("/tmp/lpool_pp.csv",
               stringsAsFactors = FALSE)

# select only relevant columns
lr <- dplyr::select(lr, price_paid, property_type, new_build, saon, paon, street,
                   locality, town, postcode)

# add concatenated column for geocoding
lr.geocode <- mutate(lr, address = paste(saon, paon, street, locality, town, postcode))

# drop previous columns
lr.geocode <- dplyr::select(lr.geocode, -saon, -paon, -street,
                           -locality, -town, -postcode)

# take sample to prevent API query limit from Google (2,500)
lr.geocode.sample <- lr.geocode[sample(nrow(lr.geocode), 2450),]

# loop over address, get lat/lon

```



```

geo.done <- data.frame()
for (i in 1:nrow(lr.geocode.sample)) {
  g <- geocode(lr.geocode.sample$address[i])
  g <- mutate(g, price=lr.geocode.sample$price_paid[i])
  geo.done <- rbind(geo.done, g)
}

# export to csv for backup
write.csv(geo.done, file = "/Users/samcomber/Documents/spatial_analysis/data/
          geocoded_lr.csv", row.names=FALSE)

# convert to spatial points df

# each property has an LSOA and an MSOA code, along with IMD info (POINT IN POLYGON)

```

$$Price_{i,j} = \beta$$

2.2 Analysis

2.3 Interpretation

3 Bibliography

4 Appendix

Avery and Clark. 2015. “R: A Language and Environment for Statistical Computing.” Journal Article. <http://www.R-project.org>.

Fotheringham and Wong. 2015. “R: A Language and Environment for Statistical Computing.” Journal Article. <http://www.R-project.org>.

Openshaw. 2015. “R: A Language and Environment for Statistical Computing.” Journal Article. <http://www.R-project.org>.