Geographically Weighted Regression (GWR)

rahmaanisa@apps.ipb.ac.id
Global Vs Local Statistics

**Global**
- similarities across space
- single-valued statistics
- non-mappable
- search for regularities
- aspatial

**Local**
- differences across space
- multi-valued statistics
- mappable
- search for exceptions
- spatial

Local statistics are spatial disaggregations of global statistics.
Introduction

Regression models are typically “global”. In some cases it can make sense to fit more flexible “local” models.

In a spatial context local refers to location. Rather than fitting a single regression model, it is possible to fit several models, one for each location (out of possibly very many) locations.
GEOGRAPHICALLY WEIGHTED REGRESSION (GWR)

GWR builds on traditional linear regression methods by permitting the relationships between variables to vary spatially.
How it works...

• By allowing the linear predictor to be a function of the spatial coordinates \((u, v)\):

\[
\eta = \sum_{j=1}^{p+1} \beta_j (u, v) x_j
\]
How it works...

• The contribution of sample points in the local regression model are weighted according to their proximity to $y_i$ by a weight function.

• A common choice is the Gaussian curve:

$$w_{ij} = \exp \left[ - \frac{d_{ij}^2}{2b^2} \right]$$

$d_{ij}$ is the distance between observation point $i$ and the realisation points $j$, and $b$ is a parameter (the bandwidth) to be determined.
Previous example

Spatially Disaggregated
IN GWR WE CAN ALSO...

• estimate local standard errors
• calculate local leverage measures
• perform tests to assess the significance of the spatial variation in the local parameter estimates
• perform tests to determine if the local model performs better than the global one
Practical Session

• There is “standalone” GWR software that can be obtained from http://ncg.nuim.ie/ncg/GWR/.

• There are various packages which will carry out GWR in R:
  • GWmodel
  • spgwr
  • gwrr
Load the package and the data set:

```r
#install.packages("GWmodel")
> library(GWmodel)

> data(LondonHP)
> help(LondonHP)
```
London Data Set

In this practical we will be working with house price data, obtained from the Nationwide Building Society (NBS) in England - this is a sample of houses sold in 2001 with mortgages arranged by NBS in and around London. The data is in a `SpatialPolygonsDataFrame` frame called `londonhp`.

```r
> head(data.frame(londonhp))
```

<table>
<thead>
<tr>
<th>PURCHASE</th>
<th>FLOORSZ</th>
<th>TYPEDETCH</th>
<th>TFSEMITDC</th>
<th>TYPETRRD</th>
<th>TYPEBNGLM</th>
<th>TYPEFLAT</th>
<th>BLDFW1</th>
<th>BLDFOSTW</th>
<th>BLD60S</th>
<th>BLD70S</th>
<th>BLD80S</th>
<th>BLD90S</th>
<th>BATH2</th>
<th>BEDS2</th>
<th>GARAGE1</th>
<th>CENTHEAT</th>
<th>UNEMPLOY</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>157000</td>
<td>77</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0.03566768</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>113500</td>
<td>75</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0.03566768</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>81750</td>
<td>64</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0.03566768</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>150000</td>
<td>95</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0.03566768</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>190000</td>
<td>107</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.03566768</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>159950</td>
<td>100</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.02408854</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PROF</th>
<th>BLDINTW</th>
<th>coords.x1</th>
<th>coords.x2</th>
<th>optional</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.4786992</td>
<td>533200</td>
<td>159400</td>
<td>TRUE</td>
</tr>
<tr>
<td>1</td>
<td>0.4786992</td>
<td>533300</td>
<td>159700</td>
<td>TRUE</td>
</tr>
<tr>
<td>2</td>
<td>0.4786992</td>
<td>532000</td>
<td>159800</td>
<td>TRUE</td>
</tr>
<tr>
<td>3</td>
<td>0.4786992</td>
<td>531900</td>
<td>160100</td>
<td>TRUE</td>
</tr>
<tr>
<td>4</td>
<td>0.4786992</td>
<td>532800</td>
<td>160300</td>
<td>TRUE</td>
</tr>
<tr>
<td>5</td>
<td>0.4773718</td>
<td>535700</td>
<td>161700</td>
<td>TRUE</td>
</tr>
</tbody>
</table>
Data Exploration

• Here, the \( y \)-variable will be housing cost per square meter of floor area:

  > londonhp$PPSQM <- londonhp$PURCHASE / londonhp$FLOORSZ

• Looking at the distribution of the price per square meter can be done using the standard \texttt{hist} command in R:

  > hist(londonhp$PPSQM, main="Price per Square Meter (Pounds)", xlab="Cost per Sq. Meter", ylab='Frequency')
Data Exploration

From this there is a fairly clear distribution shape with a small number of very costly properties (in terms of cost per unit of floor area).

Summary statistics:

\[
\text{mean(londonhp$PPSQM)} \\
[1] 1779.542
\]

\[
\text{sd(londonhp$PPSQM)} \\
[1] 657.2502
\]
Data Exploration

hist(londonhp$PROF, main="Proportion of Workforce in Professional occupations", xlab="Proportion", ylab='Frequency')
Relationship in the data

linmod <- lm(PPSQM~PROF, data=londonhp)

summary(linmod)

plot(PPSQM~PROF,data=londonhp,xlab='Proportion Professional/ Managerial', ylab='Cost per Square Metre')

abline(linmod)
library(nortest)
library(car)
library(DescTools)
library(lmtest)
ad.test(err.ols)
RunsTest(err.ols)
OLS - Model Diagnostics

moran.test(err.ols, ww, randomisation=T, alternative="greater")
bptest(linmod)
Geographically Weighted Regression (GWR)

data(LondonBorough)
plot(londonborough)
plot(londonhp, pch=16, col='firebrick',add=TRUE)

The data are rather clustered, but it may still be useful to calibrate the GWR model over a regular grid of observations.
Geographically Weighted Regression (GWR)

```r
grd <- SpatialGrid(GridTopology(c(503400,155400),
                            c(1000,1000),c(60,48)))
plot(grd)
plot(londonborough,add=TRUE,
    col=adjustcolor('navyblue',alpha.f=0.5))
```
Compute the distances between the points on the grid

```r
DM <- gw.dist(dp.locat=coordinates(londonhp),
               rp.locat=coordinates(grd))
```

```
> dim(DM)
[1] 316 2880
> head(DM[,1:10])
[1,]  52316.73  51753.65  51203.91  50667.94  50146.19  49639.10  49147.13  48670.73  48210.37  47766.52
[2,]  52127.73  51560.64  51006.86  50466.82  49940.97  49429.75  48933.63  48453.07  47988.54  47540.51
[3,]  51310.04  50759.43  50222.70  49700.30  49192.68  48700.31  48223.65  47763.17  47319.34  46892.64
[4,]  51005.29  50453.34  49915.33  49391.70  48882.92  48389.46  47911.79  47450.40  47005.74  46578.32
[5,]  51349.49  50783.56  50231.17  49692.76  49168.79  48659.74  48166.07  47688.26  47226.79  46782.15
[6,]  51959.41  51343.74  50740.32  50149.58  49571.97  49007.96  48458.02  47922.65  47402.32  46897.55
```
Basic GWR Analysis by Grid Cell

gwr.res <- gwr.basic(PPSQM~PROF, data=londonhp, 
  regression.points=grd,bw=10000, dMat=DM,kernel='gaussian')
gwr.res

Package GWmodel

Program starts at: 2017-12-18 14:15:47

Call:
gwr.basic(formula = PPSQM ~ PROF, data = londonhp, regression.points = grd,
bw = 10000, kernel = "gaussian", dMat = DM)

Dependent (y) variable: PPSQM
Independent variables: PROF
Number of data points: 316

Results of Geographically Weighted Regression

Results of Global Regression

Call:
  lm(formula = formula, data = data)

Residuals:
Min  1Q Median  3Q Max
-1079.6 -300.1 -50.6 214.5 3656.7

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  315.4      128.9   2.447  0.0149 *
PROF         3446.1     294.4  11.704  <2e-16 ***

---Significance stars
Signif. codes:  0 '****' 0.001 ***' 0.01 '**' 0.05 '*' 0.1 ' ' 1

Residual standard error: 549.3 on 314 degrees of freedom
Multiple R-squared: 0.3038
Adjusted R-squared: 0.3015
F-statistic: 137 on 1 and 314 DF,  p-value: < 2.2e-16

Results of Geographically Weighted Regression

Results of Global Regression

Program stops at: 2017-12-18 14:15:48
Automatic Bandwidth Selection

using `bw.gwr` function

```
bw.gwr(formula, data, approach="CV", kernel="bisquare",
        adaptive=FALSE, p=2, theta=0, longlat=F,dMat)
```
GWR without creating grid

DM<-gw.dist(dp.locat=coordinates(londonhp))

bw1<-bw.gwr(PURCHASE~FLOORSZ, data=londonhp, kernel = "gaussian", dMat=DM)

gwr.res1<-gwr.basic(PURCHASE~FLOORSZ, data=londonhp, bw=bw1,kernel = "gaussian", dMat=DM)
Program starts at: 2017-12-18 14:35:35

Call:
gwr.basic(formula = PURCHASE ~ FLOORSZ, data = londonhp, bw = bw1, 
kernel = "gaussian", dMat = DM)

Dependent (y) variable: PURCHASE

Independent variables: FLOORSZ

Number of data points: 316

Results of Geographically Weighted Regression

Model calibration information

Kernel function: gaussian

Fixed bandwidth: 2478.467

Regression points: the same locations as observations are used.

Distance metric: A distance matrix is specified for this model calibration

Summary of GWR coefficient estimates:

Min. 1st Qu.  Median 3rd Qu. Max.
Intercept -111786.89 -1152.31 11053.63 29696.14 131098.1
FLOORSZ 903.71 1057.27 1406.95 1917.70 3340.3

Diagnostic information

Number of data points: 316

Effective number of parameters (2trace(S) - trace(S'S)): 66.69847

Effective degrees of freedom (n-2trace(S) + trace(S'S)): 249.3015


Residual sum of squares: 252739595761

R-square value: 0.8558681

Adjusted R-square value: 0.8209574

Program stops at: 2017-12-18 14:35:35
Automatic Bandwidth Selection

DM <- gw.dist(dp.locat = coordinates(londonhp))
bw1 <- bw.gwr(PURCHASE ~ FLOORSZ, data = londonhp, kernel = "gaussian", dMat = DM)
gwr.res1 <- gwr.basic(PURCHASE ~ FLOORSZ, data = londonhp, bw = bw1, kernel = "gaussian", dMat = DM)
Package: GWmodel

Program starts at: 2017-12-18 14:35:35

Call:
gwr.basic(formula = PURCHASE ~ FLOORSZ, data = londonhp, bw = bw1, kernel = "gaussian", dMat = DM)

Dependent (y) variable: PURCHASE
Independent variables: FLOORSZ
Number of data points: 316

Results of Geographically Weighted Regression

Model calibration information

Kernel function: gaussian
Fixed bandwidth: 2478.467
Regression points: the same locations as observations are used.
Distance metric: A distance matrix is specified for this model calibration

Summary of GWR coefficient estimates:

<table>
<thead>
<tr>
<th></th>
<th>Min</th>
<th>1st Qu.</th>
<th>Median</th>
<th>3rd Qu.</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>-111786.89</td>
<td>-1152.21</td>
<td>11053.62</td>
<td>29696.14</td>
<td>131098.1</td>
</tr>
<tr>
<td>FLOORSZ</td>
<td>803.71</td>
<td>1057.27</td>
<td>1406.95</td>
<td>1917.70</td>
<td>3340.3</td>
</tr>
</tbody>
</table>

Diagnostic information

Number of data points: 316
Effective number of parameters (2trace(S) - trace(S'S)): 66.69847
Effective degrees of freedom (n-2trace(S) + trace(S'S)): 249.3015
Residual sum of squares: 252739595761
R-square value: 0.85588681
Adjusted R-square value: 0.8209574

Program stops at: 2017-12-18 14:35:35
Automatic Bandwidth Selection

bw2<-bw.gwr(PURCHASE~FLOORSZ, approach="aic", adaptive=TRUE, data=londonhp, 
   kernel = "gaussian", dMat=DM)

gwr.res2<-gwr.basic(PURCHASE~FLOORSZ, data=londonhp, 
   bw=bw2, adaptive=TRUE, kernel = "gaussian", dMat=DM)

gwr.res2
The pseudo t-tests of basic GWR outputs

gwr.t.adjust(gwr.res1)
Prediction

gwr.pred<-gwr.predict(PURCHASE~FLOORSZ, data=londonhp, bw=2000,kernel = "gaussian")
gwr.pred
References

Brunsdon, C. 2015. Geographically Weighted Regression. https://rstudio-pubs-static.s3.amazonaws.com/176883_06a3fa1fc77444be85e94dcd97ba9a34.html [18 Desember 2017]