R: Spatial regression

Roger Bivand

Thursday, 5 September 2019, 15:12-15:36

Required current contributed CRAN packages:

I am running R 3.6.1, with recent `update.packages()`.

```
needed <- c("MatrixModels", "lme4", "spatialreg", "spdep", "sf", "sp", "HSAR")
```

Beijing data set

Lower level point support data and upper level district boundaries (polygon support)

```
library(HSAR)
library(sp)
data(landSPDF)
data(landprice)
data(Beijingdistricts)
```

Convert to **sf** class and merge data with point geometries

```
library(sf)
```

```
## Linking to GEOS 3.7.2, GDAL 3.0.1, PROJ 6.2.0
```

```
land_sf <- st_as_sf(landSPDF)
landprice_sf <- merge(land_sf, landprice, by="obs")
(landprice_sf <- landprice_sf[order(landprice_sf$district.id.x),])
```

```
## Simple feature collection with 1117 features and 12 fields
## geometry type:  POINT
## dimension:      XY
## bbox:           xmin: 428553.1 ymin: 4406815 xmax: 463693.2 ymax: 4440423
## epsg (SRID):    NA
## proj4string:    +proj=tmerc +lat_0=0 +lon_0=117 +k=1 +x_0=500000 +y_0=0 +ellps=krass +units=m +no_de
## First 10 features:
##      obs district.id.x lnprice   lnarea  lndcbd dsubway   dpark    dele
## 187  189             3 5.57430 10.27820 9.94866 6.83023 7.06579 6.81916
## 188  190             3 7.16382 11.58780 9.93534 7.14334 6.78243 6.67827
## 700  968             3 7.61282  8.94551 9.91779 7.64360 6.84364 4.60356
## 701  969             3 6.81564  5.81928 9.91940 7.64640 6.88254 4.10025
## 702  970             3 6.93528  7.71869 9.91752 7.65810 6.86760 4.53460
## 709  992             3 7.45757  9.20029 9.84785 7.78904 6.95662 7.05138
## 710  993             3 7.12569  7.97788 9.84388 7.81991 7.00792 7.11267
## 711  994             3 7.48522  7.78634 9.84203 7.83398 7.03089 7.13981
## 717 1001             3 5.87349 10.70910 9.95534 7.89121 7.11019 5.67984
## 181  183             5 6.79302  6.39403 9.92025 6.76006 6.23524 6.10494
##       popden crimerate district.id.y year                 geometry
## 187 0.548966  10.75110             3    1   POINT (430237 4422804)
## 188 0.548966  10.75110             3    1 POINT (430547.1 4423001)
## 700 0.548966  10.75110             3    0 POINT (431029.4 4423667)
## 701 0.548966  10.75110             3    0 POINT (431001.8 4423695)
```

```
## 702 0.548966  10.75110             3    0 POINT (431040.9 4423697)
## 709 0.548966  10.75110             3    0   POINT (432164 4422080)
## 710 0.548966  10.75110             3    0 POINT (432239.7 4422081)
## 711 0.548966  10.75110             3    0   POINT (432275 4422082)
## 717 0.548966  10.75110             3    0 POINT (430436.8 4424601)
## 181 1.407250   2.25832             5    1 POINT (430606.5 4420186)
```

Check that the input IDs match and that the data are correctly ordered

```r
all.equal(landprice_sf$district.id.x, landprice_sf$district.id.y)
```

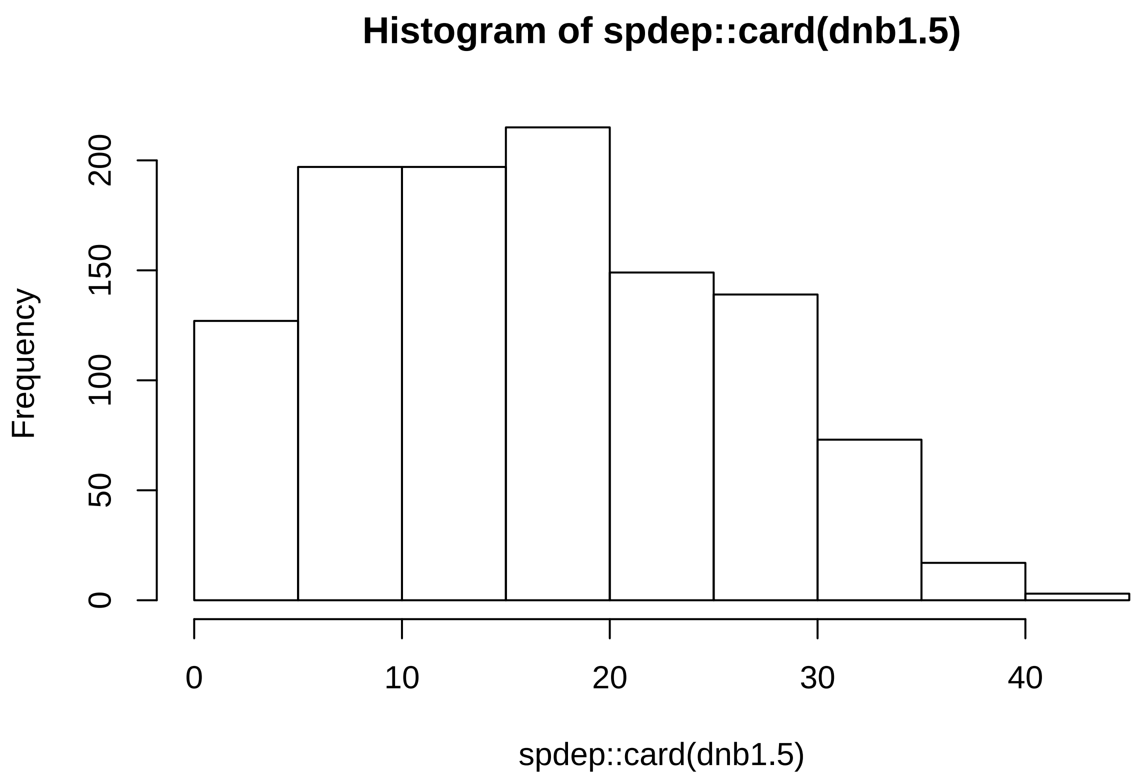
```
## [1] TRUE
```

Create the original 1.5 km distance threshold spatial weights, with a few no-neighbour observations (so set zero policy option)

```r
library(spatialreg)
dnb1.5 <- spdep::dnearneigh(landprice_sf, 0, 1500, row.names=as.character(landprice_sf$obs))
dnb1.5
```

```
## Neighbour list object:
## Number of regions: 1117
## Number of nonzero links: 18798
## Percentage nonzero weights: 1.506625
## Average number of links: 16.82901
## 7 regions with no links:
## 517 53 292 1764 33 1785 1126
```

```r
dists <- spdep::nbdists(dnb1.5, st_geometry(landprice_sf))
edists <- lapply(dists, function(x) exp((-((x/1000)^2))/(1.5^2)))
ozpo <- spdep::set.ZeroPolicyOption(TRUE)
oo <- set.ZeroPolicyOption(TRUE)
lw <- spdep::nb2listw(dnb1.5, glist=edists, style="W")
hist(spdep::card(dnb1.5))
```



Reconstruct the input data for R formula use (do not log in advance, do use factors for categorical variables to permit automatic generation of dummies)

```
landprice_sf$fyear <- factor(landprice_sf$year + 2003)
landprice_sf$price <- exp(landprice_sf$lnprice)
landprice_sf$area <- exp(landprice_sf$lnarea)
landprice_sf$Dcbd <- exp(landprice_sf$lndcbd)
landprice_sf$Dsubway <- exp(landprice_sf$dsubway)
landprice_sf$Dpark <- exp(landprice_sf$dpark)
landprice_sf$Dele <- exp(landprice_sf$dele)
landprice_sf$f_district.id <- factor(landprice_sf$district.id.x)
(t1 <- table(table(landprice_sf$f_district.id)))
```

```
##
##  1  2  3  4  5  6  7  8  9 10 11 12 14 15 16 17 18 19 20 21 23 25 26 27 28
##  7  8 10  4  5 10  8 10  8  4  6  2  4  3  1  2  5  2  1  1  1  1  1  1  2
## 31 32 33 52
##  1  1  1  1
```

Some covariates are observed at the district level rather than the land parcel level

```
sapply(as.data.frame(landprice_sf[, c("price", "area", "Dcbd", "Dele", "Dpark", "Dsubway", "crimerate",
```

```
##      price       area       Dcbd       Dele      Dpark    Dsubway crimerate
##       1062       1098       1114       1116       1114       1117        105
##     popden   geometry
##        111       1117
```

Check the matching of district IDs and counts of land parcels in districts

```
Beijingdistricts$id1 <- Beijingdistricts$id+1
all.equal(unique(landprice_sf$district.id.x), Beijingdistricts$id1)
```

```
## [1] TRUE
```

```r
(Beijingdistricts_sf <- st_as_sf(Beijingdistricts))
```

```
## Simple feature collection with 111 features and 2 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: 426987.3 ymin: 4403559 xmax: 467920.9 ymax: 4443287
## epsg (SRID):    NA
## proj4string:    +proj=tmerc +lat_0=0 +lon_0=117 +k=1 +x_0=500000 +y_0=0 +ellps=krass +units=m +no_de
## First 10 features:
##   id id1                       geometry
## 0  2   3 MULTIPOLYGON (((428183.1 44...
## 1  4   5 MULTIPOLYGON (((432472.2 44...
## 2  6   7 MULTIPOLYGON (((432446.1 44...
## 3  7   8 MULTIPOLYGON (((433534.5 44...
## 4  8   9 MULTIPOLYGON (((443807.9 44...
## 5  9  10 MULTIPOLYGON (((444461 4420...
## 6 10  11 MULTIPOLYGON (((447530.6 44...
## 7 11  12 MULTIPOLYGON (((443849.6 44...
## 8 12  13 MULTIPOLYGON (((446810 4417...
## 9 13  14 MULTIPOLYGON (((445954.8 44...
```

```r
Beijingdistricts_sf$counts <- sapply(st_contains(Beijingdistricts_sf, landprice_sf), length)
```

Check point counts by district from input data and topological points in polygon counts

```r
t2 <- table(Beijingdistricts_sf$counts)
all.equal(t1, t2)
```

```
## [1] TRUE
```

Basic formula object from the original paper and examples in package, fit and display OLS model (note that fyear is split into dummies with 2003 in the intercept)

```r
form <- log(price) ~ log(area) + log(Dcbd) + log(Dele) + log(Dpark) + log(Dsubway) +
  crimerate + popden + fyear
OLS <- lm(form, data=landprice_sf)
summary(OLS)
```

```
##
## Call:
## lm(formula = form, data = landprice_sf)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.5915 -0.5752 -0.0496  0.5206  3.4042
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  14.153917   0.370103  38.243  < 2e-16 ***
## log(area)    -0.008253   0.018675  -0.442  0.65863
## log(Dcbd)    -0.250601   0.047752  -5.248 1.84e-07 ***
## log(Dele)    -0.085528   0.032308  -2.647  0.00823 **
## log(Dpark)   -0.284372   0.046046  -6.176 9.24e-10 ***
## log(Dsubway) -0.245748   0.034755  -7.071 2.73e-12 ***
## crimerate     0.007668   0.004458   1.720  0.08575 .
```

```
## popden        0.032827   0.010304   3.186  0.00148 **
## fyear2004    -0.164503   0.058380  -2.818  0.00492 **
## fyear2005     0.017635   0.124986   0.141  0.88782
## fyear2006    -0.120314   0.107209  -1.122  0.26201
## fyear2007     0.551384   0.117431   4.695 3.00e-06 ***
## fyear2008     0.396172   0.129571   3.058  0.00229 **
## fyear2009     2.113691   0.228688   9.243  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8335 on 1103 degrees of freedom
## Multiple R-squared:  0.3524, Adjusted R-squared:  0.3448
## F-statistic: 46.17 on 13 and 1103 DF,  p-value: < 2.2e-16
```

Are the residuals spatially autocorrelated?

```
spdep::lm.morantest(OLS, listw=lw)
```

```
##
##  Global Moran I for regression residuals
##
## data:
## model: lm(formula = form, data = landprice_sf)
## weights: lw
##
## Moran I statistic standard deviate = 15.1, p-value < 2.2e-16
## alternative hypothesis: greater
## sample estimates:
## Observed Moran I      Expectation         Variance
##     0.1944768641    -0.0054494313     0.0001753028
```

What do the robust LM tests say?

```
spdep::lm.LMtests(OLS, listw=lw, test=c("RLMerr", "RLMlag"))
```

```
##
##  Lagrange multiplier diagnostics for spatial dependence
##
## data:
## model: lm(formula = form, data = landprice_sf)
## weights: lw
##
## RLMerr = 118.22, df = 1, p-value < 2.2e-16
##
##
##  Lagrange multiplier diagnostics for spatial dependence
##
## data:
## model: lm(formula = form, data = landprice_sf)
## weights: lw
##
## RLMlag = 3.7725, df = 1, p-value = 0.0521
```

How dow we do with a linear model including selected spatially lagged covariates?

```
SLX <- lmSLX(form, data=landprice_sf, listw=lw, Durbin= ~ log(area) + log(Dcbd) + log(Dele) + log(Dpark
summary(impacts(SLX))
```

```
## Impact measures (SLX, estimable, n-k):
##                      Direct    Indirect        Total
## log(area)    -0.028250900  0.15061221  0.122361308
## log(Dcbd)    -0.575325229  0.34199016 -0.233335072
## log(Dele)     0.001927612 -0.15153784 -0.149610227
## log(Dpark)   -0.066376027 -0.29670998 -0.363086011
## log(Dsubway) -0.180677830 -0.09171154 -0.272389368
## crimerate     0.002228943  0.00697053  0.009199473
## popden        0.003782010  0.04501340  0.048795406
## fyear2004    -0.163634606          NA -0.163634606
## fyear2005     0.006112745          NA  0.006112745
## fyear2006    -0.120821815          NA -0.120821815
## fyear2007     0.619330756          NA  0.619330756
## fyear2008     0.444819013          NA  0.444819013
## fyear2009     2.186988035          NA  2.186988035
## ============================================================
## Standard errors:
##                     Direct   Indirect       Total
## log(area)    0.019588590 0.03882315 0.037437930
## log(Dcbd)    0.121875390 0.13150000 0.052242219
## log(Dele)    0.051081439 0.06620277 0.043070631
## log(Dpark)   0.107848818 0.12908527 0.057968077
## log(Dsubway) 0.059325583 0.07722793 0.046759734
## crimerate    0.009157083 0.01134039 0.005658334
## popden       0.017573848 0.02455712 0.014701241
## fyear2004    0.058433277         NA 0.058433277
## fyear2005    0.124199473         NA 0.124199473
## fyear2006    0.106956715         NA 0.106956715
## fyear2007    0.117305346         NA 0.117305346
## fyear2008    0.128990353         NA 0.128990353
## fyear2009    0.227449165         NA 0.227449165
## ============================================================
## Z-values:
##                     Direct   Indirect       Total
## log(area)    -1.44221201  3.8794431  3.26837801
## log(Dcbd)    -4.72060215  2.6006857 -4.46640817
## log(Dele)     0.03773605 -2.2889955 -3.47360195
## log(Dpark)   -0.61545437 -2.2985581 -6.26355107
## log(Dsubway) -3.04552976 -1.1875436 -5.82529759
## crimerate     0.24341193  0.6146642  1.62582705
## popden        0.21520671  1.8330078  3.31913527
## fyear2004    -2.80036674         NA -2.80036674
## fyear2005     0.04921716         NA  0.04921716
## fyear2006    -1.12963282         NA -1.12963282
## fyear2007     5.27964648         NA  5.27964648
## fyear2008     3.44846729         NA  3.44846729
## fyear2009     9.61528276         NA  9.61528276
##
## p-values:
##              Direct     Indirect  Total
## log(area)    0.14924257 0.0001047 0.00108166
## log(Dcbd)    2.3515e-06 0.0093038 7.9544e-06
## log(Dele)    0.96989813 0.0220796 0.00051352
## log(Dpark)   0.53825469 0.0215300 3.7631e-10
```

```
## log(Dsubway) 0.00232271 0.2350133 5.7011e-09
## crimerate    0.80768630 0.5387765 0.10398645
## popden       0.82960616 0.0668014 0.00090297
## fyear2004    0.00510446 NA        0.00510446
## fyear2005    0.96074624 NA        0.96074624
## fyear2006    0.25863098 NA        0.25863098
## fyear2007    1.2943e-07 NA        1.2943e-07
## fyear2008    0.00056378 NA        0.00056378
## fyear2009    < 2.22e-16 NA        < 2.22e-16
```

And the spatial residual autocorrelation?

```r
spdep::lm.morantest(SLX, listw=lw)
```

```
##
##  Global Moran I for regression residuals
##
## data:
## model: lm(formula = formula(paste("y ~ ", paste(colnames(x)[-1],
## collapse = "+"))), data = as.data.frame(x), weights = weights)
## weights: lw
##
## Moran I statistic standard deviate = 14.802, p-value < 2.2e-16
## alternative hypothesis: greater
## sample estimates:
## Observed Moran I      Expectation         Variance
##     0.1885391149    -0.0069886190     0.0001745023
```

And robust LM tests?

```r
spdep::lm.LMtests(SLX, listw=lw, test=c("RLMerr", "RLMlag"))
```

```
##
##  Lagrange multiplier diagnostics for spatial dependence
##
## data:
## model: lm(formula = formula(paste("y ~ ", paste(colnames(x)[-1],
## collapse = "+"))), data = as.data.frame(x), weights = weights)
## weights: lw
##
## RLMerr = 39.985, df = 1, p-value = 2.56e-10
##
##
##  Lagrange multiplier diagnostics for spatial dependence
##
## data:
## model: lm(formula = formula(paste("y ~ ", paste(colnames(x)[-1],
## collapse = "+"))), data = as.data.frame(x), weights = weights)
## weights: lw
##
## RLMlag = 0.17014, df = 1, p-value = 0.68
```

So let's fit a spatial Durbin error model, with the same selection of spatially lagged covariates

```r
e <- eigenw(lw)
SDEM <- errorsarlm(form, data=landprice_sf, listw=lw, Durbin= ~ log(area) + log(Dcbd) + log(Dele) + log
summary(impacts(SDEM))
```

```
## Impact measures (SDEM, estimable, n):
##                    Direct     Indirect        Total
## log(area)    -0.021085385  0.133094132  0.112008747
## log(Dcbd)    -0.572352697  0.317284147 -0.255068551
## log(Dele)    -0.005661913 -0.114782232 -0.120444145
## log(Dpark)   -0.074378191 -0.273739346 -0.348117537
## log(Dsubway) -0.164798714 -0.090024131 -0.254822845
## crimerate     0.006411748 -0.001121658  0.005290091
## popden        0.004983237  0.045377975  0.050361213
## fyear2004    -0.213875338           NA -0.213875338
## fyear2005    -0.074548268           NA -0.074548268
## fyear2006    -0.136022976           NA -0.136022976
## fyear2007     0.713777760           NA  0.713777760
## fyear2008     0.488689750           NA  0.488689750
## fyear2009     2.167398316           NA  2.167398316
## ========================================================
## Standard errors:
##                   Direct   Indirect       Total
## log(area)    0.018070856 0.05086004 0.055146985
## log(Dcbd)    0.124751995 0.14355527 0.091948225
## log(Dele)    0.045634066 0.08014074 0.072364276
## log(Dpark)   0.101149512 0.14212885 0.094724658
## log(Dsubway) 0.054362580 0.09395633 0.078343698
## crimerate    0.008305157 0.01274632 0.009474872
## popden       0.015899732 0.02876860 0.023482266
## fyear2004    0.055676729         NA 0.055676729
## fyear2005    0.116369514         NA 0.116369514
## fyear2006    0.100092117         NA 0.100092117
## fyear2007    0.115930158         NA 0.115930158
## fyear2008    0.122087311         NA 0.122087311
## fyear2009    0.205699716         NA 0.205699716
## ========================================================
## Z-values:
##                  Direct    Indirect      Total
## log(area)    -1.1668171  2.61687048  2.0310947
## log(Dcbd)    -4.5879242  2.21018802 -2.7740454
## log(Dele)    -0.1240721 -1.43225824 -1.6644144
## log(Dpark)   -0.7353292 -1.92599427 -3.6750467
## log(Dsubway) -3.0314734 -0.95814867 -3.2526272
## crimerate     0.7720201 -0.08799852  0.5583285
## popden        0.3134164  1.57734403  2.1446488
## fyear2004    -3.8413776          NA -3.8413776
## fyear2005    -0.6406168          NA -0.6406168
## fyear2006    -1.3589779          NA -1.3589779
## fyear2007     6.1569636          NA  6.1569636
## fyear2008     4.0027890          NA  4.0027890
## fyear2009    10.5367103          NA 10.5367103
##
## p-values:
##              Direct     Indirect Total
## log(area)    0.24328423 0.008874 0.04224539
## log(Dcbd)    4.4768e-06 0.027092 0.00553640
## log(Dele)    0.90125821 0.152070 0.09602964
## log(Dpark)   0.46213902 0.054105 0.00023781
```

```
## log(Dsubway) 0.00243363 0.337988 0.00114343
## crimerate    0.44010252 0.929878 0.57662011
## popden       0.75396431 0.114716 0.03198094
## fyear2004    0.00012235 NA       0.00012235
## fyear2005    0.52177167 NA       0.52177167
## fyear2006    0.17415359 NA       0.17415359
## fyear2007    7.4153e-10 NA       7.4153e-10
## fyear2008    6.2600e-05 NA       6.2600e-05
## fyear2009    < 2.22e-16 NA       < 2.22e-16
```

The likelihood ratio test shows that the SDEM model fits much better than the SLX model

```r
LR1.sarlm(SDEM)
```

```
##
##  Likelihood Ratio diagnostics for spatial dependence
##
## data:
## Likelihood ratio = 117.78, df = 1, p-value < 2.2e-16
## sample estimates:
## Log likelihood of spatial error model
##                             -1299.824
##           Log likelihood of OLS fit y
##                             -1358.713
```

The Hausman test is perhaps significant, suggestion that the non-spatial coefficients shift somewhat between the SLX model and the SDEM model

```r
Hausman.test(SDEM)
```

```
##
##  Spatial Hausman test (asymptotic)
##
## data:  NULL
## Hausman test = 41.781, df = 21, p-value = 0.004482
```

But …

is this the end of the story? Reach out to very general mixed model IID random effects at the district level (fixed effects would give 111 dummies); here without spatially lagged covariates

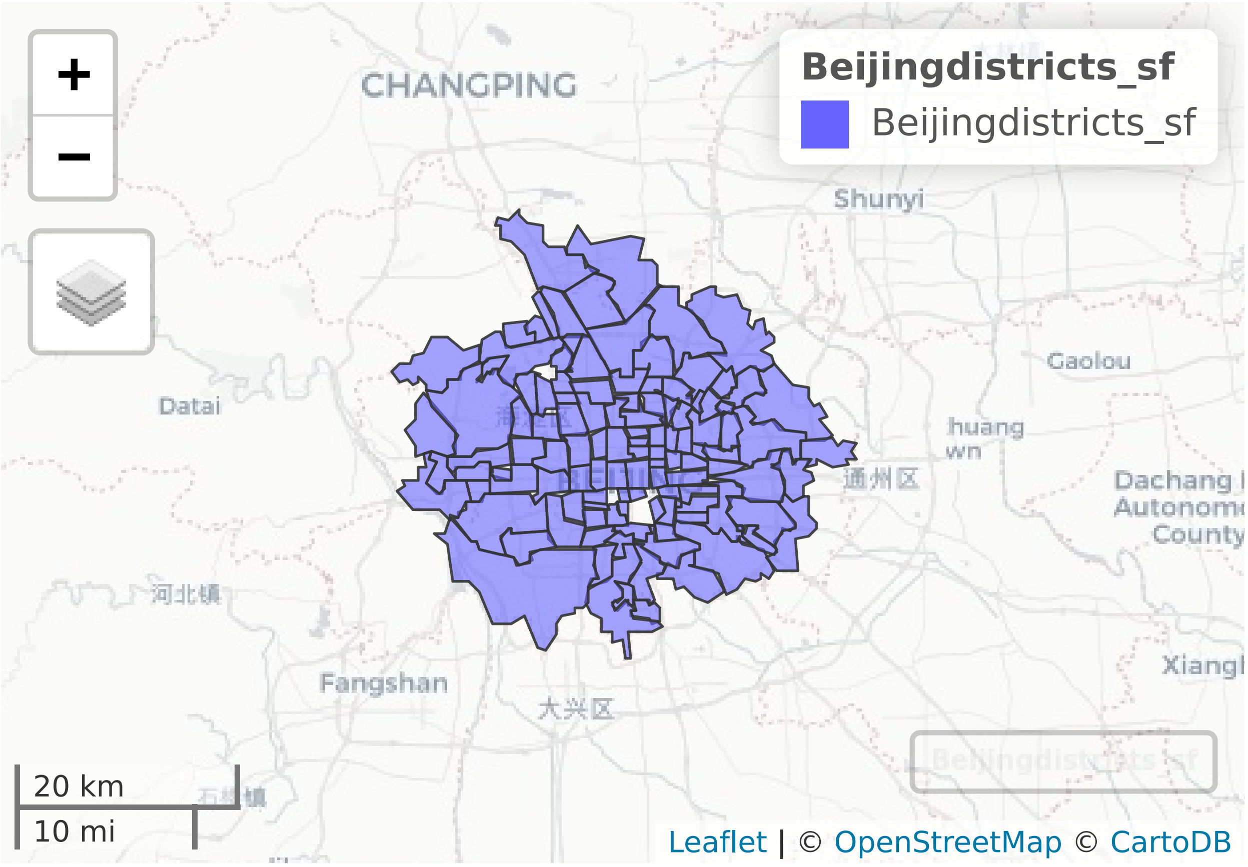
```r
library(lme4)
mlm_1 <- lmer(update(form, . ~ . + (1 | f_district.id)), data=landprice_sf, REML=FALSE)
Beijingdistricts_sf$mlm_re <- ranef(mlm_1)[[1]][,1]
```

The **HSAR** model gives a spatial error model at the district level, defining a sparse matrix `Delta` assigning parcels to districts

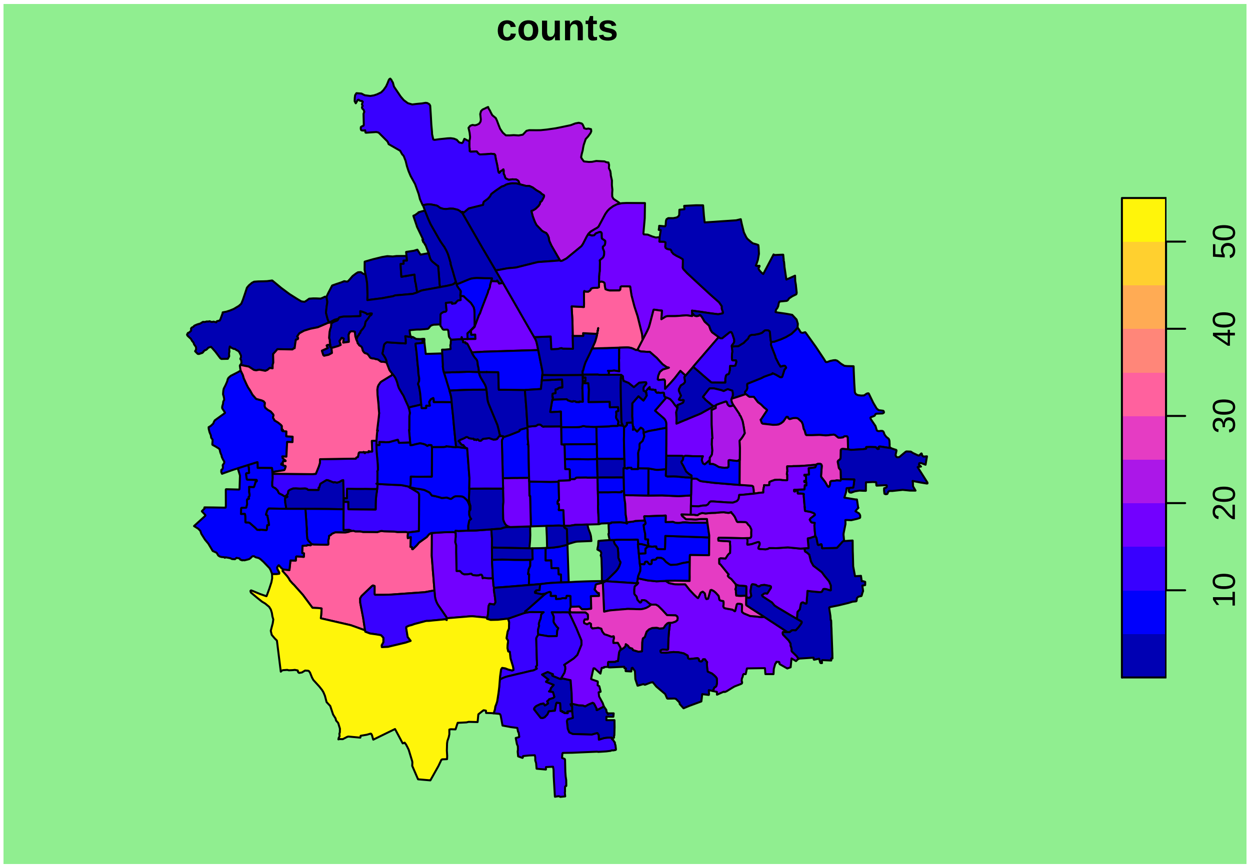
```r
library(Matrix)
suppressMessages(library(MatrixModels))
Delta <- as(model.Matrix(~ -1 + f_district.id, data=landprice_sf, sparse=TRUE), "dgCMatrix")
```

There are gaps in the land parcel and district coverage

```r
library(mapview)
mapview(Beijingdistricts_sf)
```




```
opar <- par(bg="lightgreen")
plot(Beijingdistricts_sf[, "counts"])
```



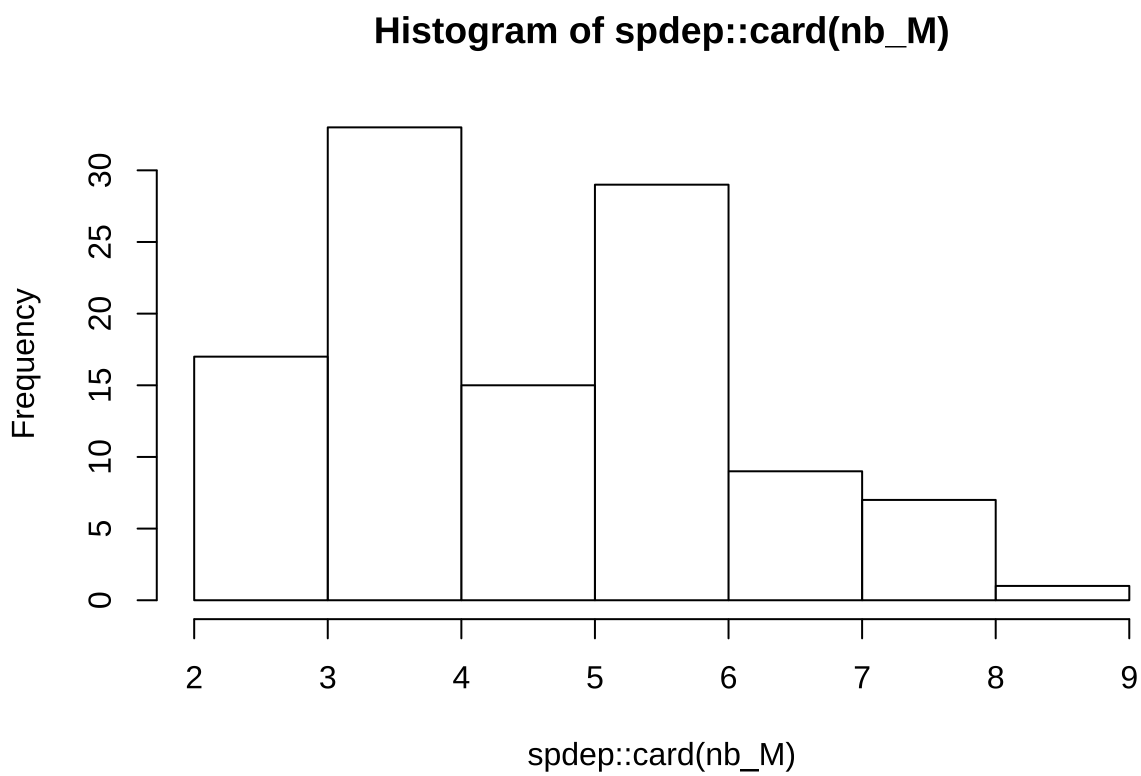

```
par(opar)
```

Construct the spatial weights for the disticts

```
nb_M <- spdep::poly2nb(Beijingdistricts, queen=FALSE, row.names=as.character(Beijingdistricts$id1))
M <- as(spdep::nb2listw(nb_M, style="B"), "CsparseMatrix")
dim(M)
```

```
## [1] 111 111
```

```
hist(spdep::card(nb_M))
```

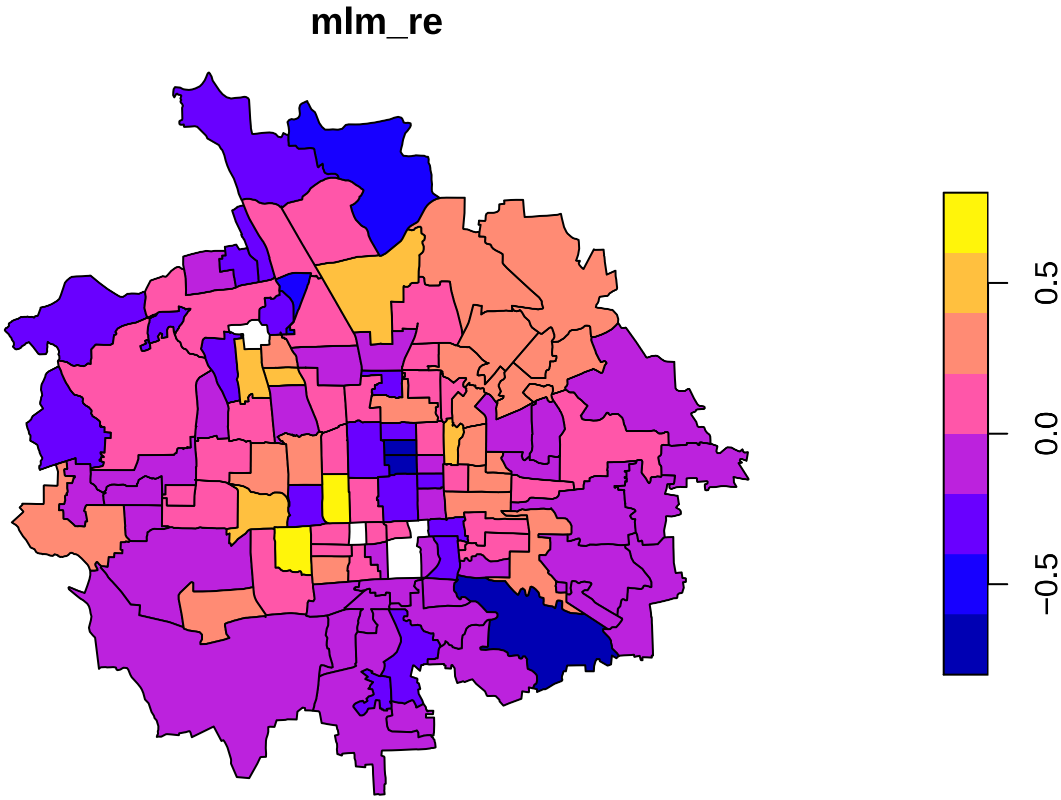


Using the M sparse spatial weights matrix, fit a model with district level simultaneous error autoregression; without spatially lagged covariates

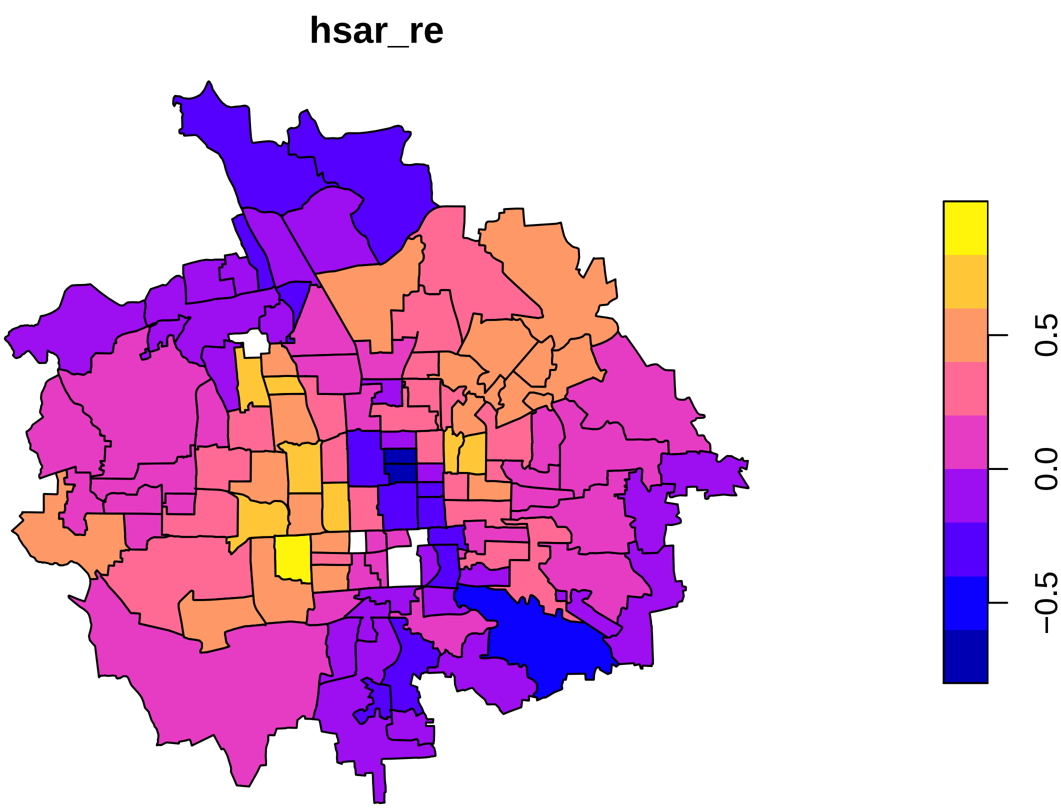
```
m_hsar <- hsar(form, data=landprice_sf, W=NULL, M=M, Delta=Delta, burnin=500, Nsim=5000, thinning=1)
Beijingdistricts_sf$hsar_re <- m_hsar$Mus[1,]
```

The IID and SAR random effects are rather similar

```
plot(Beijingdistricts_sf[,"mlm_re"])
```




```r
plot(Beijingdistricts_sf[,"hsar_re"])
```



We do not have tests for residual autocorrelation for these fitted multilevel models, so (speculatively) let's copy out the district level random effects to the parcels, checking first for matching

```
o <- match(landprice_sf$district.id.x, Beijingdistricts_sf$id1)
landprice_sf$id1 <- Beijingdistricts_sf$id1[o]
all.equal(landprice_sf$district.id.x, landprice_sf$id1)
```

```
## [1] TRUE
```

```
landprice_sf$mlm_re <- Beijingdistricts_sf$mlm_re[o]
landprice_sf$hsar_re <- Beijingdistricts_sf$hsar_re[o]
```

Now we can refit the SLX model but including the district level IID random effect, and test the residual autocorrelation

```
spdep::lm.morantest(lmSLX(update(form, . ~ . + mlm_re), data=landprice_sf, listw=lw, Durbin= ~ log(area
```

```
##
##  Global Moran I for regression residuals
##
## data:
## model: lm(formula = formula(paste("y ~ ", paste(colnames(x)[-1],
## collapse = "+"))), data = as.data.frame(x), weights = weights)
## weights: lw
##
## Moran I statistic standard deviate = 2.7727, p-value = 0.002779
## alternative hypothesis: greater
## sample estimates:
## Observed Moran I      Expectation         Variance
##      0.028860257     -0.007654903      0.000173432
```

and for the spatially structured random effect

```
spdep::lm.morantest(lmSLX(update(form, . ~ . + hsar_re), data=landprice_sf, listw=lw, Durbin= ~ log(area
```

```
##
##  Global Moran I for regression residuals
##
## data:
## model: lm(formula = formula(paste("y ~ ", paste(colnames(x)[-1],
## collapse = "+"))), data = as.data.frame(x), weights = weights)
## weights: lw
##
## Moran I statistic standard deviate = 3.3167, p-value = 0.0004554
## alternative hypothesis: greater
## sample estimates:
## Observed Moran I      Expectation         Variance
##     0.0359635684    -0.0077079227     0.0001733743
```

Next fit SDEM models with the IID random effect, and it turns out that the SLX model does almost as well, so maybe most of the residual autocorrelation was at the district level rather than the parcel level?

```
SDEM1 <- errorsarlm(update(form, . ~ . + mlm_re), data=landprice_sf, listw=lw, Durbin= ~ log(area) + log
```

```
LR1.sarlm(SDEM1)
```

```
##
##  Likelihood Ratio diagnostics for spatial dependence
##
## data:
```

```
## Likelihood ratio = 4.3635, df = 1, p-value = 0.03672
## sample estimates:
## Log likelihood of spatial error model
##                             -1224.315
##             Log likelihood of OLS fit y
##                             -1226.496
```

The SSRE doesn't do as well (perhaps because it oversmooths the districts)

```
SDEM2 <- errorsarlm(update(form, . ~ . + hsar_re), data=landprice_sf, listw=lw, Durbin= ~ log(area) + lo
```

```
LR1.sarlm(SDEM2)
```

```
##
##  Likelihood Ratio diagnostics for spatial dependence
##
## data:
## Likelihood ratio = 6.6411, df = 1, p-value = 0.009965
## sample estimates:
## Log likelihood of spatial error model
##                             -1229.587
##             Log likelihood of OLS fit y
##                             -1232.908
```

Isn't spatial regression fun!!