Waller SISMID 2025: Houston Crime Slippery Slopes

Lance A. Waller

7/15/2025

Houston Crime Geographically Weighted Regression Example

This file reads in data from a shapefile of Houston census tracts (using the same code in the markdown file we used for Lecture 2).

Next we will use "spgwr" to fit geographically weighted regression to see if the associations between tract-level (standardized) alcohol sales or illegal drug arrests are correlated with tract-level violent crime rates.

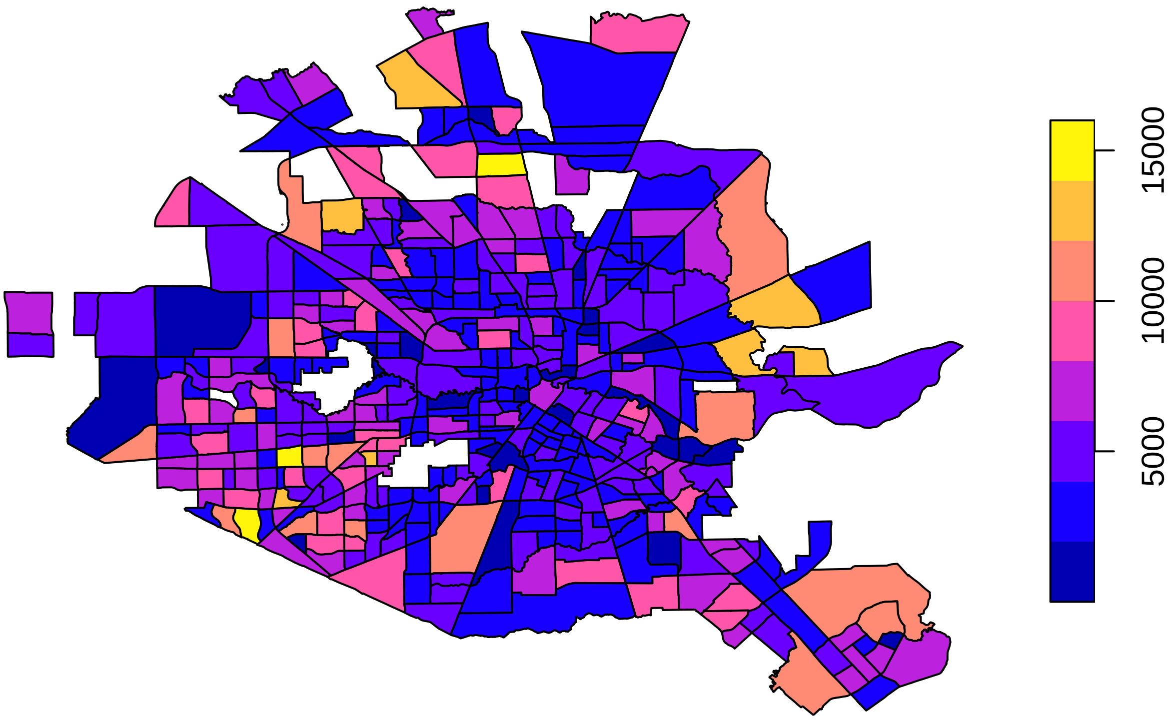
Same code as in GIS example to open libraries, set working directory, and read in the Houston shapefile.

```
##Load libraries

pacman::p_load(sf, #replaces "maptools", "rgdal" and other deprecated packages
               tmap, #helps with plotting your map
               RColorBrewer, # creates nice color schemes
               ClassInt,  # finds class intervals for continuous variables
               GWmodel, # Adds the geographically weighted regression functions
               tidyverse,
               gridExtra,
               here # For constructing filepaths relative to root directory
               )


##Read in shapefile - Houston Census Tracts
#houston = read_sf(dsn = here("data") , layer = "HoustonENAR2012final")
#***HERE***
houston = st_read(dsn = paste(path,"data/",sep=""),layer = "HoustonENAR2012final")
```

```
## Reading layer `HoustonENAR2012final' from data source
##   `/Users/lwaller/Library/CloudStorage/OneDrive-Emory/meetings/SISMID.2024/SISMID_2024_spatial_statis
##   using driver `ESRI Shapefile'
## Simple feature collection with 439 features and 133 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -95.75434 ymin: 29.52563 xmax: -95.06086 ymax: 30.03774
## CRS:           NA
```

```
##Map the tracts using plot...
plot(houston['POP2000'],main="Population 2000",
     sub="Quantile (Equal-Frequency) Class Intervals")
```

Population 2000



That's a pretty vivid color scheme, let's use tmap's tm_shape to draw nicer maps of the (standardized) violence rate, and the two covariates. (Similar code to that from Lecture 2).

======= # The data table has a lot of census data and various transformations # of the violent crime, alcohol sales, and drug arrest data. The next # section pulls the values we want.

Outcome: Number of violent crimes by tract

violence = houston$violence_2

Divide by the 2000 population to get the rate

violence.rate = violence/houston$tot_pop

Covariate 1 (log standardized total alcohol sales)

Z.log.total = houston$Zl_total

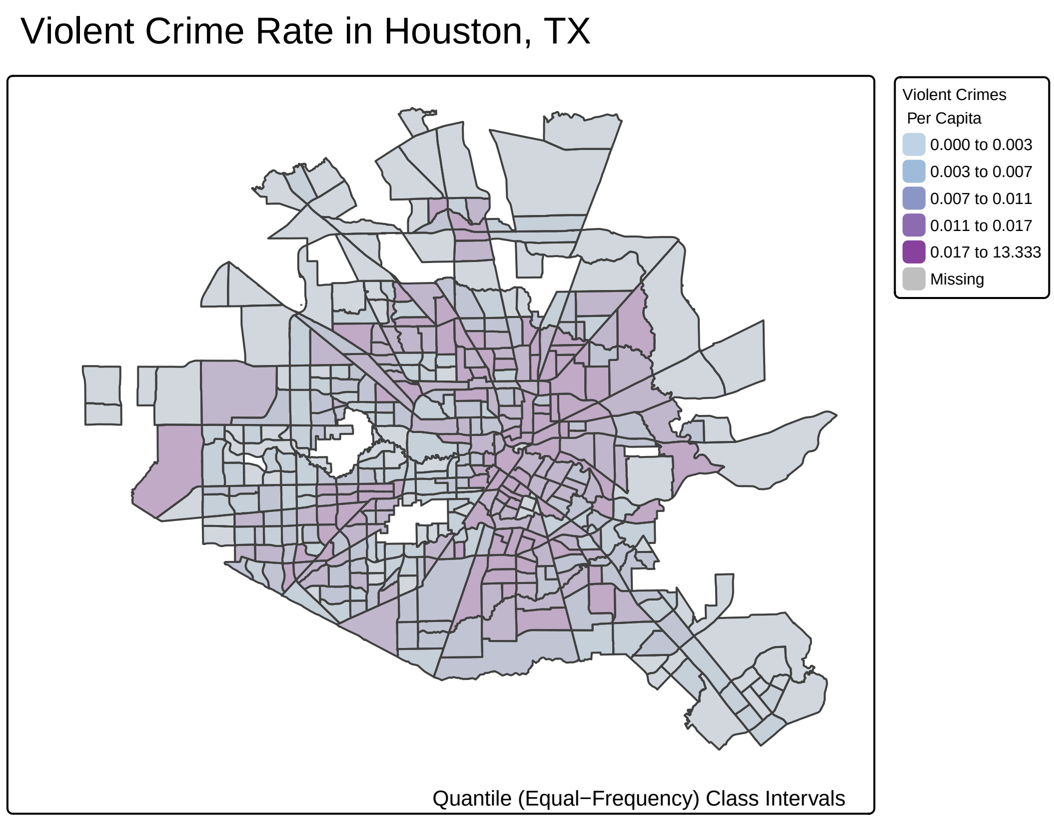
Covariate 2 (log standardized illegal drug arrests)

Z.log.drug = houston$Zl_drug

```r
# Divide by the 2000 population to get the rate
houston$violence.rate = houston$violence_2/houston$tot_pop

# Look at all 9 hex codes in BuPu
#scales::show_col(brewer.pal(9,name="BuPu"))
# Saves all 9 hex codes to a palette
all_colors <- brewer.pal(9, "BuPu")
# Subset to start from the third color
subset_colors <- all_colors[3:(1+6)]
#Display the colors
#subset_colors

violence_map <- tm_shape(houston) +
  tm_fill('violence.rate',
style='quantile',
#midpoint = median(houston$violence.rate),
palette=subset_colors,
title='Violent Crimes \n Per Capita') +
  tm_borders(alpha=0.7) +
  tm_credits('Quantile (Equal-Frequency) Class Intervals',
             position=c('RIGHT', 'BOTTOM')) +
  tm_layout(main.title="Violent Crime Rate in Houston, TX",
            inner.margins = c(0.1, 0.1, 0.05, 0.05),
            main.title.size=1.2, legend.title.size=0.5,
            legend.text.size=0.5)
```

```
##
## -- tmap v3 code detected -------------------------------------------------------
## [v3->v4] `tm_fill()`: instead of `style = "quantile"`, use fill.scale =
## `tm_scale_intervals()`.
## i Migrate the argument(s) 'style', 'palette' (rename to 'values') to
##   'tm_scale_intervals(<HERE>)'
## [v3->v4] `tm_fill()`: migrate the argument(s) related to the legend of the
## visual variable `fill` namely 'title' to 'fill.legend = tm_legend(<HERE>)'
## [v3->v4] `tm_borders()`: use 'fill' for the fill color of polygons/symbols
## (instead of 'col'), and 'col' for the outlines (instead of 'border.col').
## [v3->v4] `tm_borders()`: use `fill_alpha` instead of `alpha`.
## [v3->v4] `tm_layout()`: use `tm_title()` instead of `tm_layout(main.title = )`
```

```r
violence_map
```



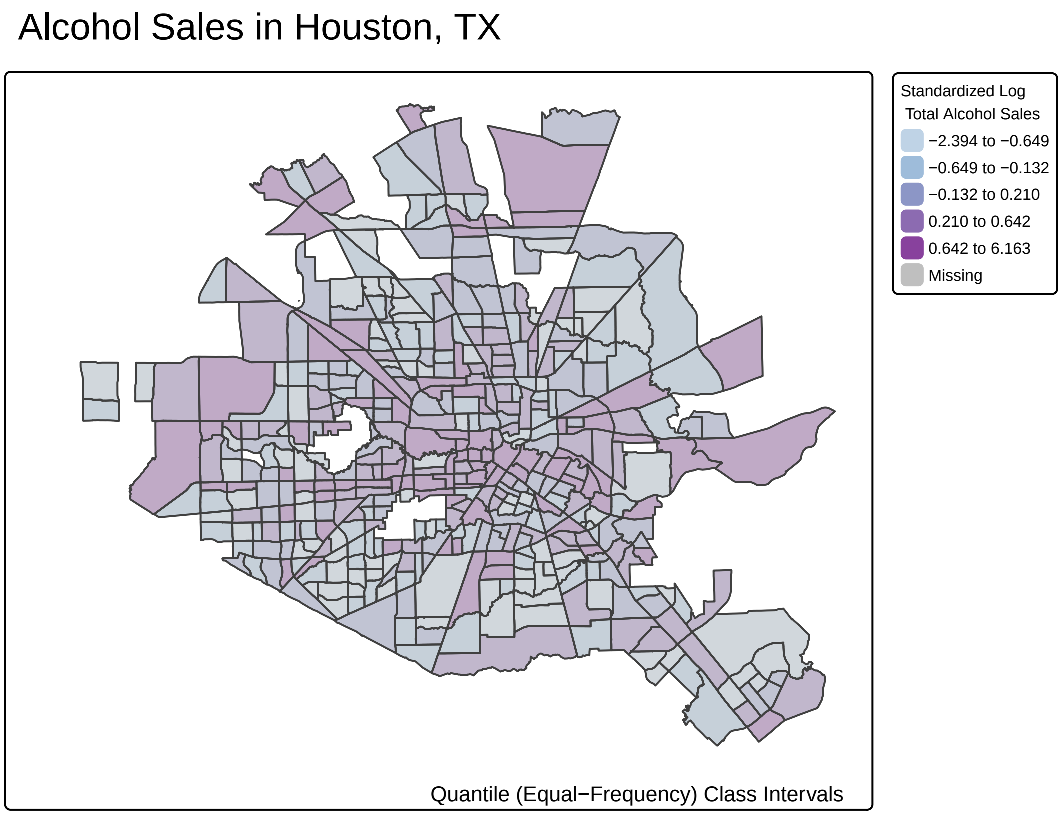
Next, map standardized log total alcohol sales.

```
alc_map <- tm_shape(houston) +
  tm_fill('Zl_total',
style='quantile',
palette=subset_colors,
midpoint = mean(houston$Zl_total),
title='Standardized Log \n Total Alcohol Sales') + # "\n" moves text to the next line
  tm_borders(alpha=0.7) +
  tm_credits('Quantile (Equal-Frequency) Class Intervals',
             position=c('RIGHT', 'BOTTOM')) +
  tm_layout(main.title="Alcohol Sales in Houston, TX",
            inner.margins = c(0.1, 0.1, 0.05, 0.05),
            main.title.size=1.2, legend.title.size=0.5,
            legend.text.size=0.5)
```

```
##
## -- tmap v3 code detected -------------------------------------------------------
## [v3->v4] `tm_fill()`: instead of `style = "quantile"`, use fill.scale =
## `tm_scale_intervals()`.
## i Migrate the argument(s) 'style', 'midpoint', 'palette' (rename to 'values')
##   to 'tm_scale_intervals(<HERE>)'
## For small multiples, specify a 'tm_scale_' for each multiple, and put them in a
## list: 'fill'.scale = list(<scale1>, <scale2>, ...)'
## [v3->v4] `tm_fill()`: migrate the argument(s) related to the legend of the
## visual variable `fill` namely 'title' to 'fill.legend = tm_legend(<HERE>)'
```

```
## [v3->v4] `tm_borders()`: use `fill_alpha` instead of `alpha`.
## [v3->v4] `tm_layout()`: use `tm_title()` instead of `tm_layout(main.title = )`
```

```r
alc_map
```



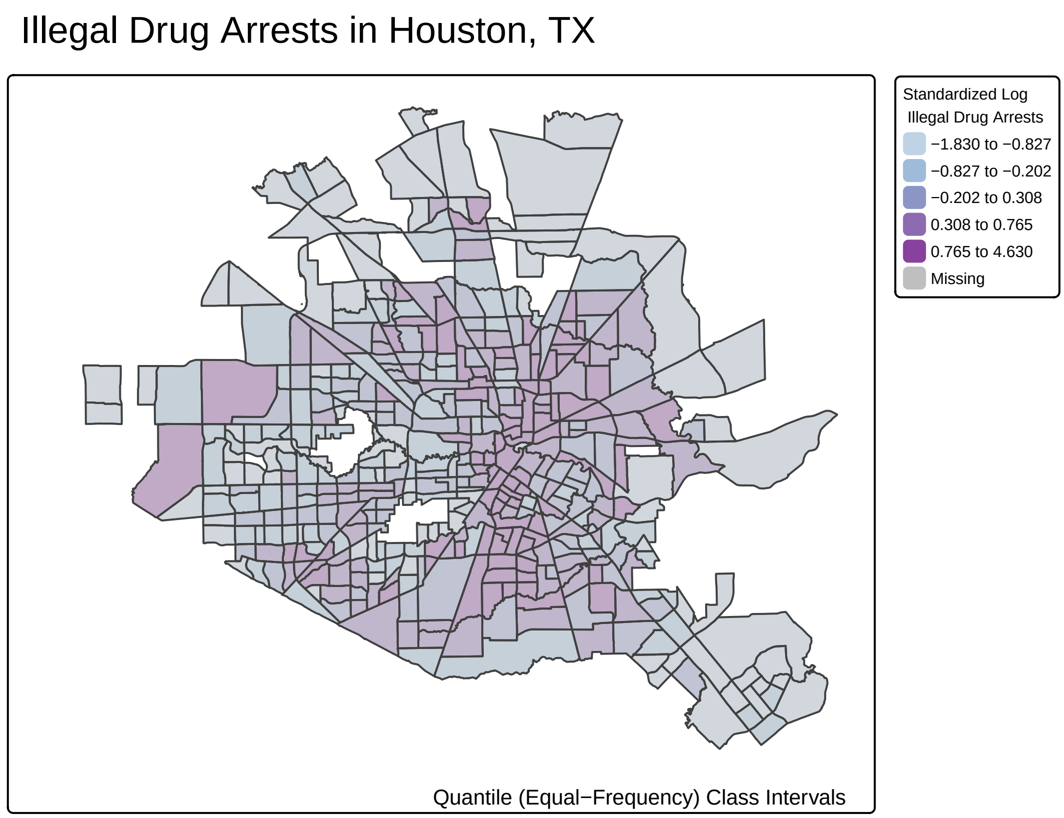
Finally, plot standardized log illegal drug arrests.

```r
drug_map <- tm_shape(houston) +
  tm_fill('Zl_drug',
style='quantile',
palette=subset_colors,
midpoint=mean(houston$Zl_drug),
title='Standardized Log \n Illegal Drug Arrests') +
  tm_borders(alpha=0.7) +
  tm_credits('Quantile (Equal-Frequency) Class Intervals',
             position=c('RIGHT', 'BOTTOM')) +
  tm_layout(main.title="Illegal Drug Arrests in Houston, TX",
            inner.margins = c(0.1, 0.1, 0.05, 0.05),
            main.title.size=1.2, legend.title.size=0.5,
            legend.text.size=0.5)
```

```
##
## -- tmap v3 code detected -------------------------------------------------------
## [v3->v4] `tm_fill()`: instead of `style = "quantile"`, use fill.scale =
## `tm_scale_intervals()`.
## i Migrate the argument(s) 'style', 'midpoint', 'palette' (rename to 'values')
##   to 'tm_scale_intervals(<HERE>)'
```

```
## For small multiples, specify a 'tm_scale_' for each multiple, and put them in a
## list: 'fill'.scale = list(<scale1>, <scale2>, ...)'
## [v3->v4] `tm_fill()`: migrate the argument(s) related to the legend of the
## visual variable `fill` namely 'title' to 'fill.legend = tm_legend(<HERE>)'
## [v3->v4] `tm_borders()`: use `fill_alpha` instead of `alpha`.
## [v3->v4] `tm_layout()`: use `tm_title()` instead of `tm_layout(main.title = )`
```

```r
drug_map
```



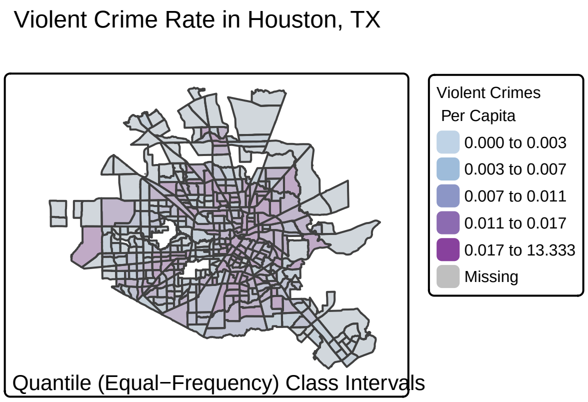
These three figures will match the maps in Figure 1 of:

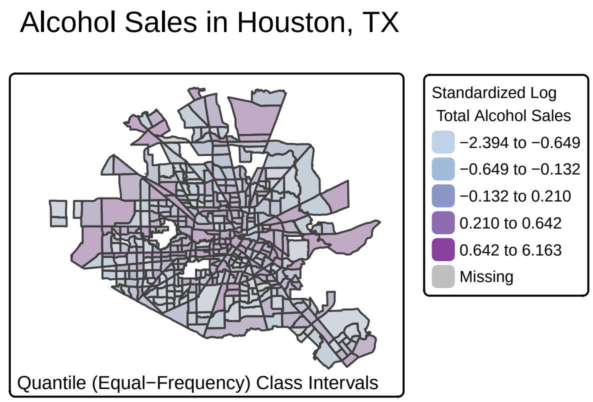
Waller LA, Zhu L, Gotway CA, Gorman DM, and Gruenewald PJ (2007) "Quantifying geographic variations in associations between alcohol distribution and violence: A comparison of geographically weighted regression and spatially varying coefficient models". Stochastic Environmental Research and Risk Assessment.21, 573-588.

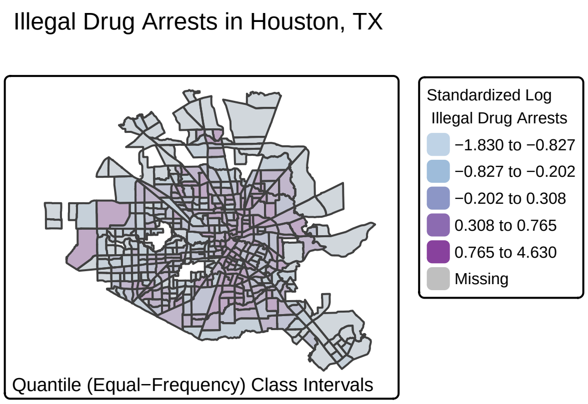
Let's use tmap to make a multiple-map single figure.

```r
tmap_arrange(violence_map, alc_map, drug_map)
```

```
## [plot mode] fit legend/component: Some legend items or map compoments do not
## fit well, and are therefore rescaled.
## i Set the tmap option `component.autoscale = FALSE` to disable rescaling.
## [plot mode] fit legend/component: Some legend items or map compoments do not
## fit well, and are therefore rescaled.
## i Set the tmap option `component.autoscale = FALSE` to disable rescaling.
## [plot mode] fit legend/component: Some legend items or map compoments do not
## fit well, and are therefore rescaled.
## i Set the tmap option `component.autoscale = FALSE` to disable rescaling.
```







Now we are ready to fit *geographically weighted regression* (GWR). GWR uses weighting to fit local regression values to allow the slopes associated with each covariate to change (smoothly) over space. (See Lecture notes for details). To estimate the association $\beta(s)$ for a location s, GWR uses kernel weights (similar to our intensity estimation for point process data) to weight observations (outcomes and covariates) near s more to give an estimate of β associated with location s. If you move over, the weights change and the estimated association (β). The kernels make sure that if you don't move far, the estimate doesn't change much so we get a smooth surface for β.

We do this in two steps.

First, we estimate the bandwidth for the kernel using cross validation (CV).

Next, we use the estimated bandwidth to fit the Poisson GWR model.

To get the bandwidth, "ggwr.sel" will test the cross validation score to find a minimum.

```
### Now to fit Poisson GWR!
# The function 'ggwr' in the 'spgwr' package uses syntax similar to 'glm'
# (like we would use in a standard Poisson regression).
# 'longlat' tells the function that our coordinates are in longitude and
# latitude coordinates.
# 'ggwr.sel' selects the bandwidth for GWR based on the data, the model,
# and cross-validation
# Create distance matrix from centroids
```

```r
houston.sp <- houston %>%
  as('Spatial')
DM <- gw.dist(dp.locat = coordinates(houston.sp))
head(houston)
```

```
## Simple feature collection with 6 features and 134 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -95.48509 ymin: 29.9511 xmax: -95.25861 ymax: 30.03774
## CRS:           NA
##       AREA        FIPS STATE_FIPS CNTY_FIPS STCOFIPS  TRACT POP2000 POP2001
## 1  1.65532 48201553200         48       201    48201 553200    7246    7319
## 2  7.00482 48201240900         48       201    48201 240900    9439    9531
## 3  6.84941 48201240700         48       201    48201 240700    3632    3810
## 4  4.56927 48201550300         48       201    48201 550300    9654   10094
## 5 21.15775 48201240300         48       201    48201 240300    3627    3678
## 6  5.69720 48201550400         48       201    48201 550400   13153   13821
##   POP00_SQMI WHITE BLACK AMERI_ES ASIAN HAWN_PI OTHER MULT_RACE HISPANIC MALES
## 1     4377.4  5368   930       30   252       6   448       212     1582  3591
## 2     1347.5  6263  1834       48   347      17   671       259     1844  4620
## 3      530.3  2106   710        9   294      17   381       115     1008  1805
## 4     2112.8  5830  2403       49   380      12   680       300     1713  4751
## 5      171.4  2095   958       21   140      39   257       117      802  1824
## 6     2308.7  4959  5467       54   751       7  1452       463     3672  6610
##   FEMALES AGE_UNDER5 AGE_5_17 AGE_18_21 AGE_22_29 AGE_30_39 AGE_40_49 AGE_50_64
## 1    3655        499     1134       474      1218      1149       924      1060
## 2    4819        736     2130       477      1000      1657      1609      1350
## 3    1827        342      739       204       595       791       545       338
## 4    4903        798     1223       602      2556      1985      1231       979
## 5    1803        284      650       286       608       610       500       453
## 6    6543       1345     2556      1022      2585      2858      1669       873
##   AGE_65_UP MED_AGE MED_AGE_M MED_AGE_F HOUSEHOLDS AVE_HH_SZ HSEHLD_1_M
## 1       788    32.2      30.8      34.2       2902      2.37        481
## 2       480    32.4      31.5      33.2       3151      2.99        197
## 3        78    29.2      29.0      29.5       1336      2.72        182
## 4       280    28.9      29.2      28.6       4871      1.98       1184
## 5       236    29.8      30.0      29.7       1474      2.45        263
## 6       245    27.4      27.1      27.6       5099      2.58        852
##   HSEHLD_1_F MARHH_CHD MARHH_NO_C MHH_CHILD FHH_CHILD FAMILIES AVE_FAM_SZ
## 1        445       519        720        68       227     1728       3.01
## 2        251      1079        904        86       237     2550       3.31
## 3        132       445        252        29       103      911       3.30
## 4        990       585        776       106       450     2175       2.82
## 5        216       284        271        53       128      867       3.17
## 6        722      1288        759       167       561     3112       3.28
##   HSE_UNITS VACANT OWNER_OCC RENTER_OCC PER_POVE PER_ASSI PER_UNEM PER_FH_C
## 1      3117    215       970       1932 11.05320  1.30788  4.28205 13.13660
## 2      3249     98      2552        599  3.84314  1.11572  3.55041  9.29412
## 3      1408     72       796        540  8.56202  1.11857  1.92698 11.30630
## 4      5380    509       552       4319  9.70115  2.14988  5.69940 20.68970
## 5      1634    160       632        842 11.30330  2.27425  4.85744 14.76360
## 6      5519    420      1660       3439  7.00514  1.47435  4.12750 18.02700
##    PER_AA PER_HISP PER_MOVE PER_OWN PER_VACA RATIO18 POP00_SQ PER_M15_   TOTAL
## 1 12.8347  21.8327  67.4052 31.1197  6.89766 3.43723   4377.4  8.14242 0.62103
```

```
## 2 19.4300  19.5360  52.7488 78.5472  3.01631 2.29344   1347.5  6.98167 0.14832
## 3 19.5485  27.7533  60.7110 56.5341  5.11364 2.35985    530.3  8.06718 0.27533
## 4 24.8912  17.7439  78.8490 10.2602  9.46097 3.77685   2112.8  8.69070 0.17609
## 5 26.4130  22.1119  60.4775 38.6781  9.79192 2.88330    171.4  8.68486 0.68928
## 6 41.5647  27.9176  77.7140 30.0779  7.61007 2.37170   2308.7  9.19182 0.12164
##        ON     OFF    BOTH    DRUG VIOLENCE   X_COORD  Y_COORD  L_POVE  L_ASSI
## 1 0.27601 0.12421 0.22081 0.00000 0.041402 -95.44610 30.02584 2.40272 0.26841
## 2 0.08475 0.02119 0.04238 0.00000 0.010594 -95.29420 30.01946 1.34629 0.10950
## 3 0.11013 0.13767 0.02753 0.02753 0.027533 -95.40499 29.99625 2.14734 0.11205
## 4 0.10358 0.06215 0.01036 0.00000 0.072509 -95.44132 29.99339 2.27224 0.76541
## 5 0.22057 0.19300 0.27571 0.00000 0.220568 -95.32401 29.98572 2.42509 0.82165
## 6 0.05322 0.03801 0.03041 0.00000 0.083631 -95.45344 29.98570 1.94664 0.38822
##    L_UNEM  L_FH_C    L_AA  L_HISP  L_MOVE L_OWNE L_VACA L_RATIO L_POP_SQ
## 1 1.45443 2.57540 2.55215 3.08341 4.21072  3.438  1.931 1.23467  8.38421
## 2 1.26706 2.22938 2.96682 2.97226 3.96554  4.364  1.104 0.83005  7.20601
## 3 0.65595 2.42536 2.97290 3.32335 4.10612  4.035  1.632 0.85860  6.27344
## 4 1.74036 3.02964 3.21451 2.87604 4.36753  2.328  2.247 1.32889  7.65577
## 5 1.58051 2.69216 3.27386 3.09612 4.10227  3.655  2.282 1.05893  5.14400
## 6 1.41767 2.89187 3.72725 3.32926 4.35304  3.404  2.029 0.86361  7.74444
##     L_M15  L_TOTAL     L_ON    L_OFF   L_BOTH   L_DRUG L_VIOLEN OID_
## 1 2.09709 -0.47638 -1.28732 -2.08578 -1.51045 -5.59943 -3.18443  438
## 2 1.94329 -1.90838 -2.46805 -3.85423 -3.16108 -5.59943 -4.54747   87
## 3 2.08780 -1.28978 -2.20609 -1.98290 -3.59248 -3.59248 -3.59237   86
## 4 2.16225 -1.73676 -2.26741 -2.77820 -4.56980 -5.59943 -2.62404  429
## 5 2.16158 -0.37211 -1.51154 -1.64507 -1.28841 -5.59943 -1.51155   83
## 6 2.21831 -2.10669 -2.93332 -3.26991 -3.49298 -5.59943 -2.48134  430
##      TRACT200 PER_POVE_1 PER_ASSI_1 PER_UNEM_1 PER_FH_C_1 PER_AA_1 PER_HISP_1
## 1 48201553200   11.05320    1.30788    4.28205   13.13660  12.8347    21.8327
## 2 48201240900    3.84314    1.11572    3.55041    9.29412  19.4300    19.5360
## 3 48201240700    8.56202    1.11857    1.92698   11.30630  19.5485    27.7533
## 4 48201550300    9.70115    2.14988    5.69940   20.68970  24.8912    17.7439
## 5 48201240300   11.30330    2.27425    4.85744   14.76360  26.4130    22.1119
## 6 48201550400    7.00514    1.47435    4.12750   18.02700  41.5647    27.9176
##   PER_MOVE_1 PER_OWN_1 PER_VACA_1 RATIO18_1 POP00_SQ_1 PER_M151 TOTAL_1    ON_1
## 1    67.4052   31.1197    6.89766   3.43723     4377.4  8.14242 0.62103 0.27601
## 2    52.7488   78.5472    3.01631   2.29344     1347.5  6.98167 0.14832 0.08475
## 3    60.7110   56.5341    5.11364   2.35985      530.3  8.06718 0.27533 0.11013
## 4    78.8490   10.2602    9.46097   3.77685     2112.8  8.69070 0.17609 0.10358
## 5    60.4775   38.6781    9.79192   2.88330      171.4  8.68486 0.68928 0.22057
## 6    77.7140   30.0779    7.61007   2.37170     2308.7  9.19182 0.12164 0.05322
##     OFF_1  BOTH_1  DRUG_1 VIOLENCE_1 X_COORD_1 Y_COORD_1 L_POVE_1 L_ASSI_1
## 1 0.12421 0.22081 0.00000   0.041402 -95.44610  30.02584  2.40272  0.26841
## 2 0.02119 0.04238 0.00000   0.010594 -95.29420  30.01946  1.34629  0.10950
## 3 0.13767 0.02753 0.02753   0.027533 -95.40499  29.99625  2.14734  0.11205
## 4 0.06215 0.01036 0.00000   0.072509 -95.44132  29.99339  2.27224  0.76541
## 5 0.19300 0.27571 0.00000   0.220568 -95.32401  29.98572  2.42509  0.82165
## 6 0.03801 0.03041 0.00000   0.083631 -95.45344  29.98570  1.94664  0.38822
##   L_UNEM_1 L_FH_C_1  L_AA_1 L_HISP_1 L_MOVE_1 L_OWNE_1 L_VACA_1 L_RATIO_1
## 1  1.45443  2.57540 2.55215  3.08341  4.21072    3.438    1.931   1.23467
## 2  1.26706  2.22938 2.96682  2.97226  3.96554    4.364    1.104   0.83005
## 3  0.65595  2.42536 2.97290  3.32335  4.10612    4.035    1.632   0.85860
## 4  1.74036  3.02964 3.21451  2.87604  4.36753    2.328    2.247   1.32889
## 5  1.58051  2.69216 3.27386  3.09612  4.10227    3.655    2.282   1.05893
## 6  1.41767  2.89187 3.72725  3.32926  4.35304    3.404    2.029   0.86361
```

```
##   L_POP_SQ_1 L_M15_1 L_TOTAL_1   L_ON_1  L_OFF_1 L_BOTH_1 L_DRUG_1 L_VIOLEN_1
## 1    8.38421 2.09709  -0.47638 -1.28732 -2.08578 -1.51045 -5.59943   -3.18443
## 2    7.20601 1.94329  -1.90838 -2.46805 -3.85423 -3.16108 -5.59943   -4.54747
## 3    6.27344 2.08780  -1.28978 -2.20609 -1.98290 -3.59248 -3.59248   -3.59237
## 4    7.65577 2.16225  -1.73676 -2.26741 -2.77820 -4.56980 -5.59943   -2.62404
## 5    5.14400 2.16158  -0.37211 -1.51154 -1.64507 -1.28841 -5.59943   -1.51155
## 6    7.74444 2.21831  -2.10669 -2.93332 -3.26991 -3.49298 -5.59943   -2.48134
##      stfid200  tract200_1 violence_2    expect tot_pop total_12 drug_12
## 1 48201553200 48201553200          3  67.56548    7246  0.62103 0.00000
## 2 48201240900 48201240900          1  88.01415    9439  0.14832 0.00000
## 3 48201240700 48201240700          1  33.86666    3632  0.27533 0.02753
## 4 48201550300 48201550300          7  90.01892    9654  0.17609 0.00000
## 5 48201240300 48201240300          8  33.82004    3627  0.68928 0.00000
## 6 48201550400 48201550400         11 122.64542   13153  0.12164 0.00000
##   l_total_12 l_drug_12  Zl_total   Zl_drug                       geometry
## 1   -0.46040  -4.60517  1.132030 -1.830210 MULTIPOLYGON (((-95.4293 30...
## 2   -1.84314  -4.60517 -0.044432 -1.830210 MULTIPOLYGON (((-95.3301 30...
## 3   -1.25411  -3.28261  0.456726 -1.035468 MULTIPOLYGON (((-95.38018 2...
## 4   -1.68152  -4.60517  0.093078 -1.830210 MULTIPOLYGON (((-95.42301 2...
## 5   -0.35770  -4.60517  1.219409 -1.830210 MULTIPOLYGON (((-95.27718 2...
## 6   -2.02768  -4.60517 -0.201442 -1.830210 MULTIPOLYGON (((-95.43658 2...
##   violence.rate
## 1  0.0004140215
## 2  0.0001059434
## 3  0.0002753304
## 4  0.0007250880
## 5  0.0022056796
## 6  0.0008363111
```

```
houston.bw <- bw.gwr(violence_2 ~ Zl_total + Zl_drug + offset(log(POP2000)),
             data = houston.sp,
             adaptive=TRUE,
             dMat=DM)
```

```
## Adaptive bandwidth: 278 CV score: 527176.2
## Adaptive bandwidth: 180 CV score: 505698
## Adaptive bandwidth: 117 CV score: 475152.8
## Adaptive bandwidth: 81 CV score: 454134.2
## Adaptive bandwidth: 55 CV score: 450084.7
## Adaptive bandwidth: 43 CV score: 455241.7
## Adaptive bandwidth: 66 CV score: 450199.1
## Adaptive bandwidth: 51 CV score: 451466.9
## Adaptive bandwidth: 60 CV score: 449677.7
## Adaptive bandwidth: 61 CV score: 449799.6
## Adaptive bandwidth: 57 CV score: 449395.6
## Adaptive bandwidth: 57 CV score: 449395.6
```

```
bgwr <- ggwr.basic(violence_2 ~ Zl_total + Zl_drug + offset(log(POP2000)),
                   data =houston.sp,
                   family = "poisson",
                   bw = houston.bw,
                   kernel = "bisquare",
                   adaptive = TRUE,
                   dMat = DM)
```

```
##  Iteration    Log-Likelihood
```

```
## ========================
##        0        -4580
##        1        -4034
##        2        -4044
##        3        -4014
##        4        -4011
##        5        -4011
##        6        -4011
```

Now to map the results.

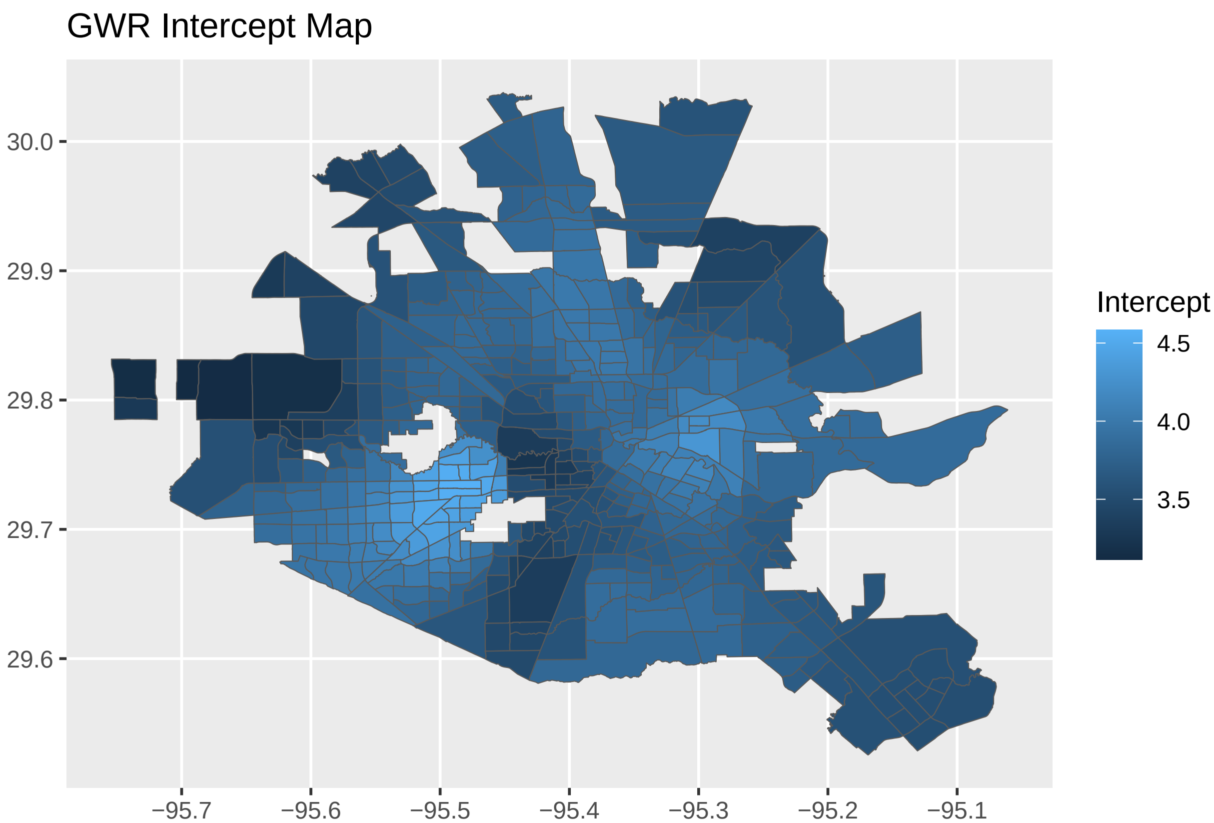
The houston.ggwr object contains an element called SDF which contains the spatial data frame (SDF) of the outcomes. We will want to map the (spatially varying) intercept and the parameters associated with each of our two covariates. The notation is tricky since the spatial data frame is within the houston.ggwr object.

The spatial data frame from ggwr is in the old "sp" spatial format, so first we convert this to the newer graphic format of "sf" (simple format) so we can use ggplot to make maps

---

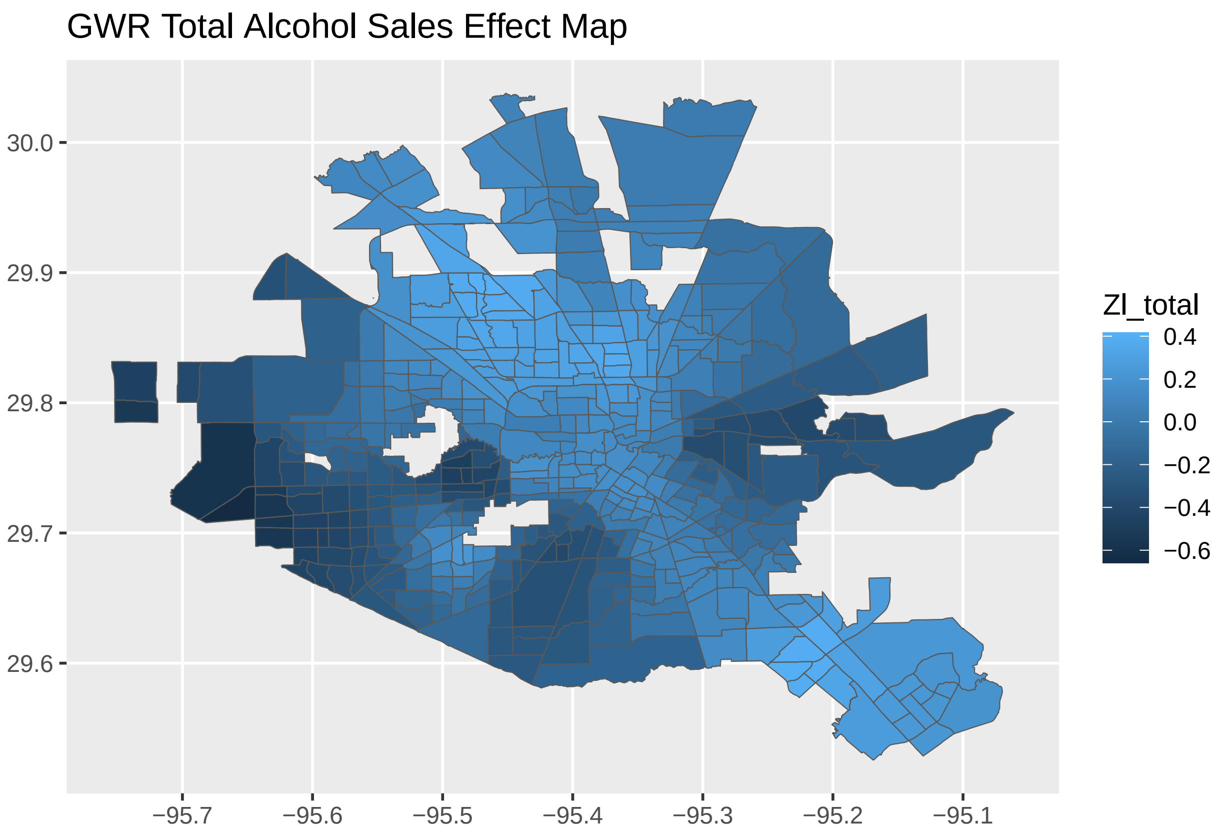
Plot the map of intercept estimates

```r
# Convert sp into sf
bgwr.sp <- bgwr$SDF
bgwr.sf <- st_as_sf(bgwr.sp)

# Map intercept using ggplot()
intercept_map <- ggplot() + geom_sf(data=bgwr.sf, aes(fill=Intercept)) +
  coord_sf() +
  ggtitle(paste("GWR Intercept Map"))

intercept_map
```

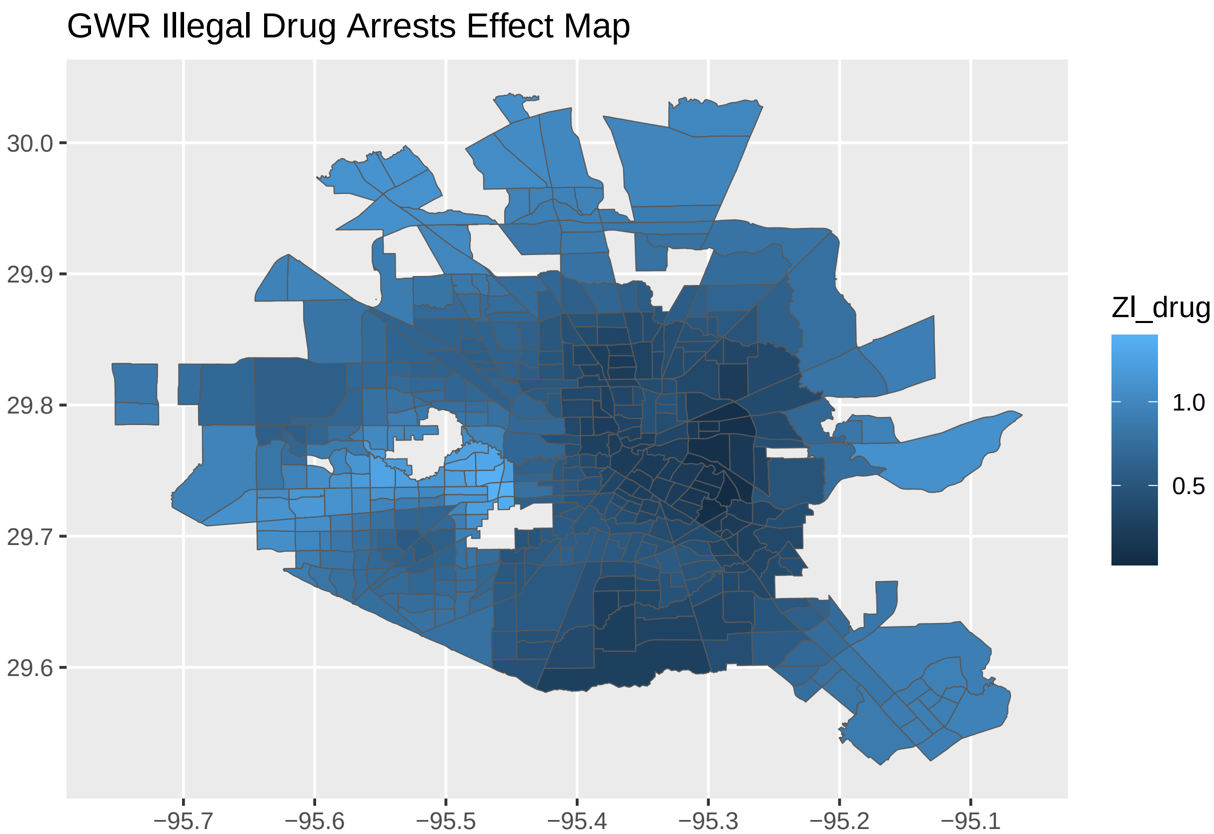


Plot the map of estimates of association with standardized total alcohol sales.

```
# Map local beta (slope) for total alcohol sales using ggplot()
beta_alcohol_map <- ggplot() + geom_sf(data=bgwr.sf, aes(fill=Zl_total)) +
  coord_sf() +
  ggtitle(paste("GWR Total Alcohol Sales Effect Map"))

beta_alcohol_map
```

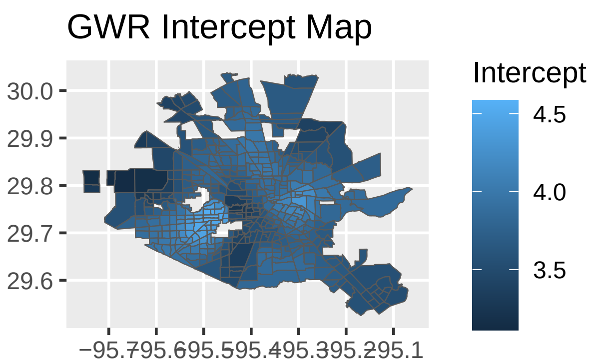


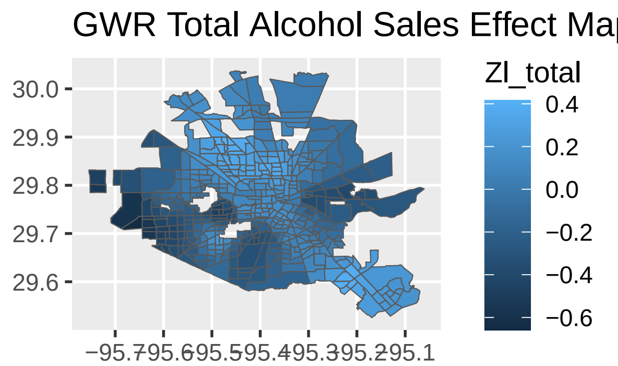

```r
# Map local beta (slope) for illegal drug arrests using ggplot()
beta_drug_map <- ggplot() + geom_sf(data=bgwr.sf, aes(fill=Zl_drug)) +
  coord_sf() +
  ggtitle(paste("GWR Illegal Drug Arrests Effect Map"))

beta_drug_map
```

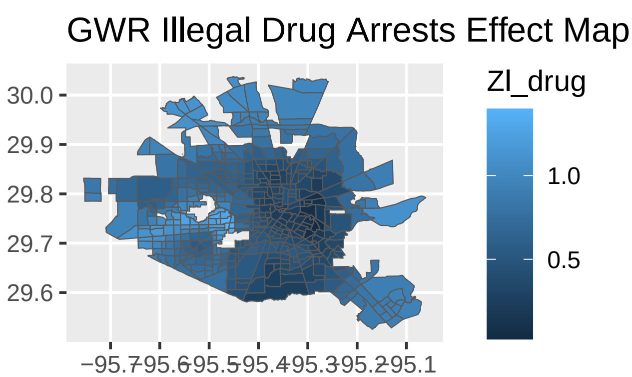


Now make multi-map figure of all three maps

```
grid.arrange(intercept_map, beta_alcohol_map, beta_drug_map, nrow=2)
```







These are very smooth maps. Let's try it with a smaller bandwidth.

```
# Try with a smaller, fixed bandwidth (1/4 the size of the other)
# This is closer to what is presented in Figure 5 of Waller et al. 2007

ggwr.bwdiv4 =ggwr.basic(violence_2 ~ Zl_total + Zl_drug + offset(log(POP2000)),
                      data =houston.sp,
                      family = "poisson",
                      bw = (houston.bw/4),
                      kernel = "bisquare",
                      adaptive = TRUE,
                      dMat = DM)
```

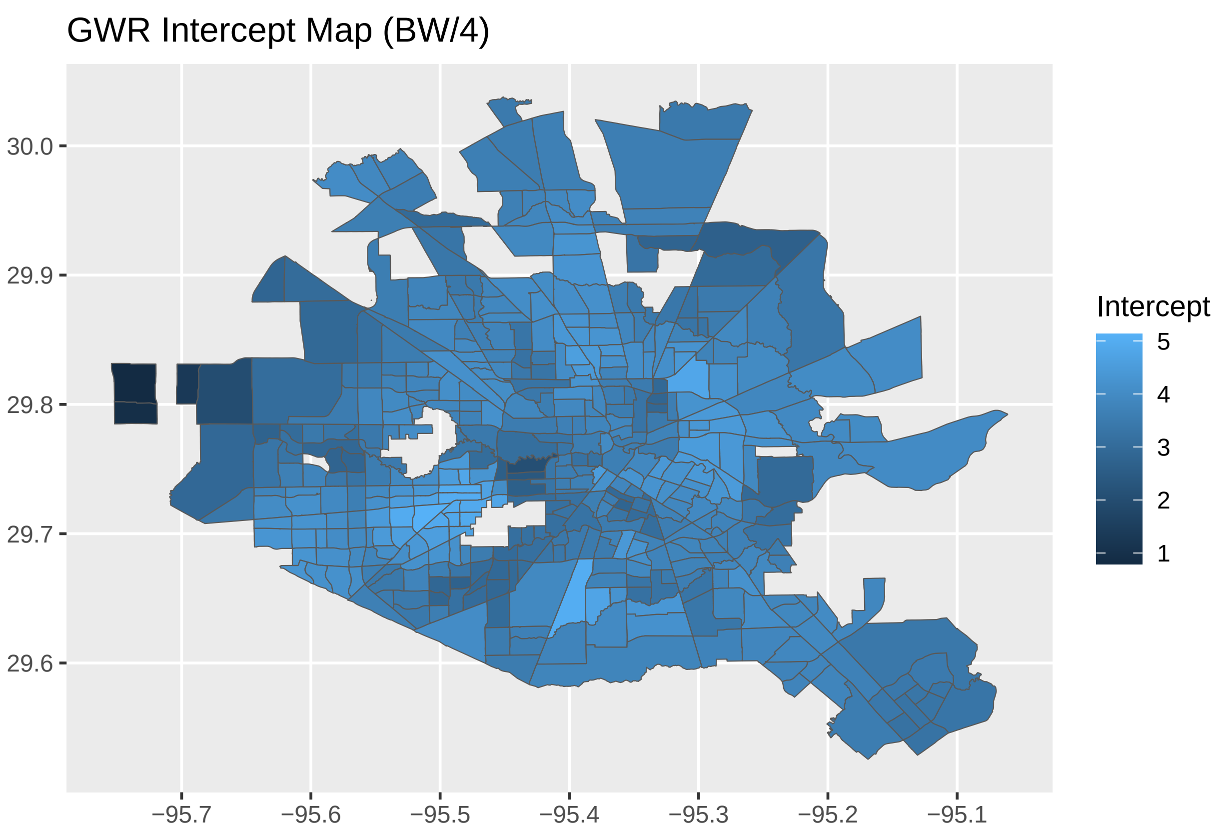
```
##  Iteration    Log-Likelihood
## =========================
##        0          -2620
##        1          -2526
##        2          -2364
##        3          -2266
##        4          -2240
##        5          -2235
##        6          -2234
##        7          -2234
##        8          -2234
```

```
# Now to map the outcomes
```

```r
# The houston.ggwr object contains an element called SDF which contains
# the spatial data frame (SDF) of the outcomes.
# We will want to map the (spatially varying) intercept and the parameters
# associated with each of our two covariates.
#  The notation is tricky since the spatial data frame is within the
# houston.ggwr object.
# ggwr$SDF@data
intercepts = ggwr.bwdiv4$SDF@data$Intercept
alcohol.effects = ggwr.bwdiv4$SDF@data$Zl_total
drug.effects = ggwr.bwdiv4$SDF@data$Zl_drug
```

Plot the map of intercept estimates

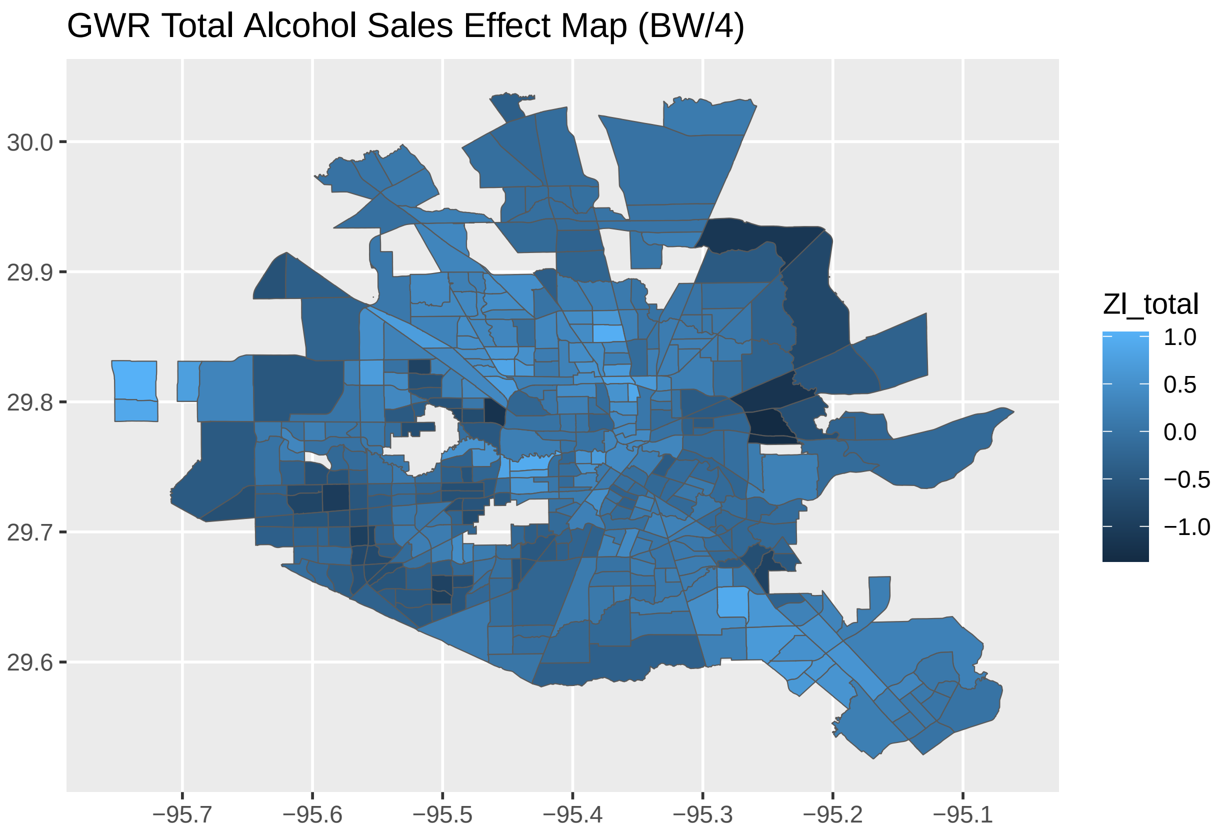
```r
# Convert sp into sf
ggwr.bwdiv4.sp <- ggwr.bwdiv4$SDF
ggwr.bwdiv4.sf <- st_as_sf(ggwr.bwdiv4.sp)

# Map intercept using ggplot()
intercept_map_bwdiv4 <- ggplot() + geom_sf(data=ggwr.bwdiv4.sf, aes(fill=Intercept)) +
  coord_sf() +
  ggtitle(paste("GWR Intercept Map (BW/4)"))

intercept_map_bwdiv4
```



Plot the map of estimates of association with standardized total alcohol sales.

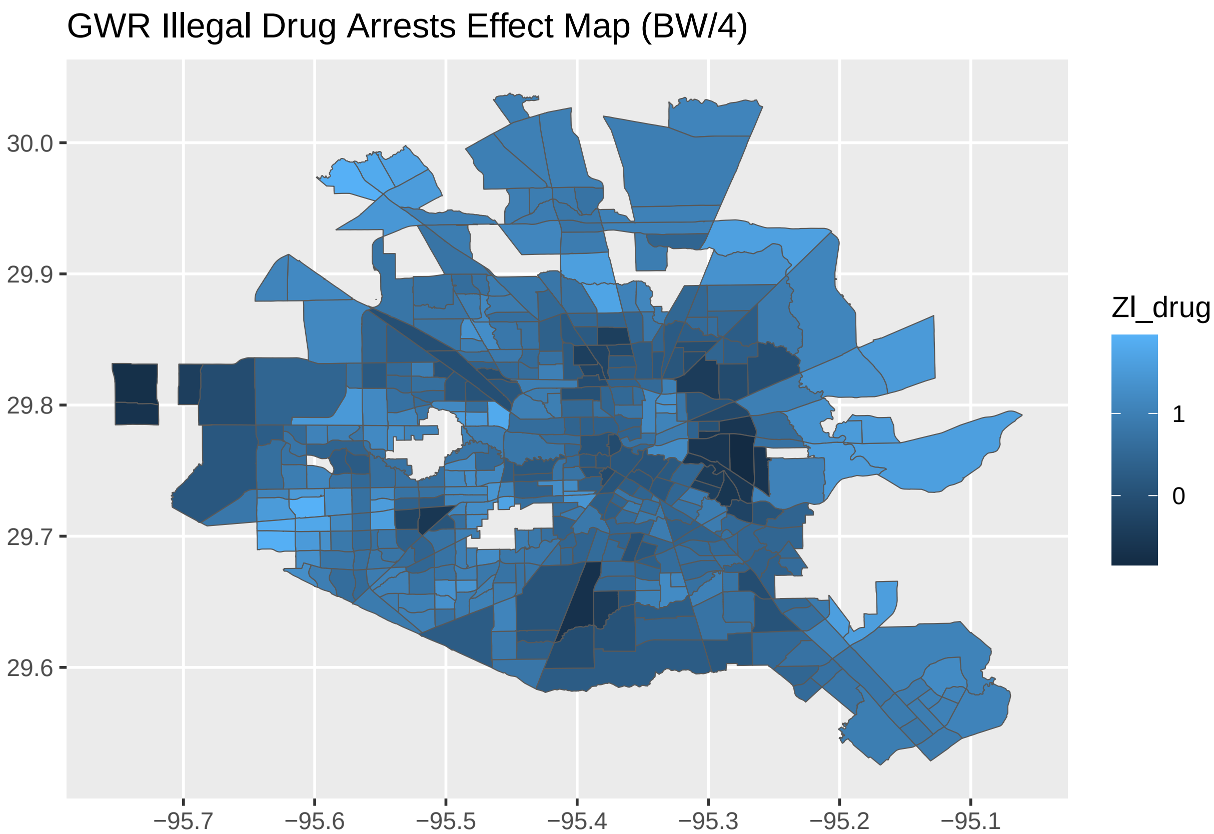
```r
# Map local beta (slope) for total alcohol sales using ggplot()
beta_alcohol_map_bwdiv4 <- ggplot() + geom_sf(data=ggwr.bwdiv4.sf, aes(fill=Zl_total)) +
  coord_sf() +
```

```
  ggtitle(paste("GWR Total Alcohol Sales Effect Map (BW/4)"))

beta_alcohol_map_bwdiv4
```



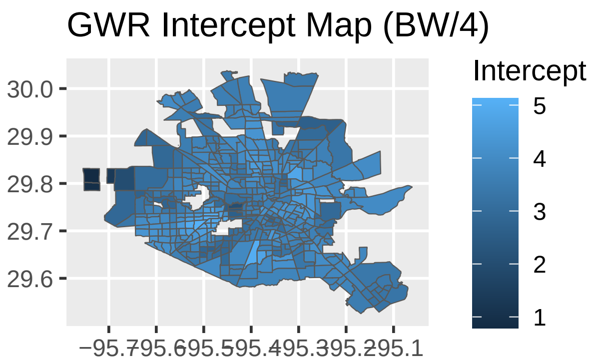
Plot the map of estimates of association with standardized illegal drug arrests

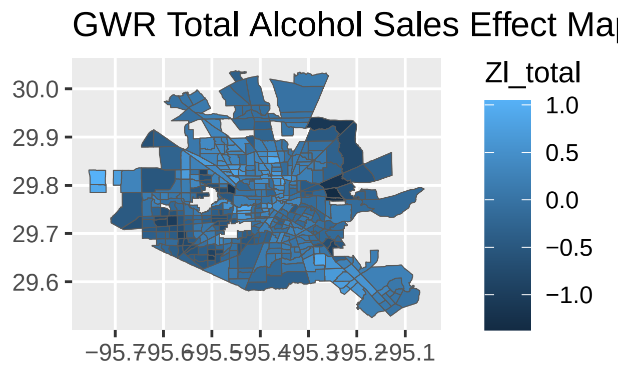
```
# Map local beta (slope) for illegal drug arrests using ggplot()
beta_drug_map_bwdiv4 <- ggplot() + geom_sf(data=ggwr.bwdiv4.sf, aes(fill=Zl_drug)) +
  coord_sf() +
  ggtitle(paste("GWR Illegal Drug Arrests Effect Map (BW/4)"))

beta_drug_map_bwdiv4
```

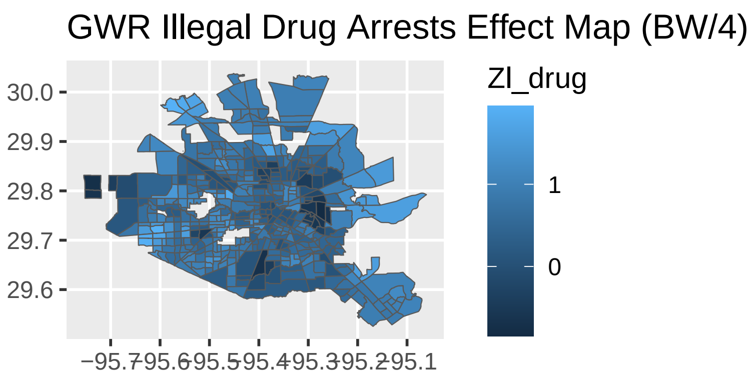


Now make multi-map figure of all three maps

```
grid.arrange(intercept_map_bwdiv4, beta_alcohol_map_bwdiv4, beta_drug_map_bwdiv4, nrow=2)
```







What do you see?

What do the maps mean? Statistically? Epidemiologically?
