A Monte Carlo Investigation of Locally Weighted Regression

Aaron Swoboda and Sam Carruthers

October 29, 2012

This document writes up the results of the recent run of `uberScript.R`. It contains the following code:

```
# set our simulation parameters
Replications = 100
sample.size = c(50, 100, 200, 500, 1000)
error.sd = c(2, 4, 6)
B1.spatial.var = c(0, .1, .2, .3)
B2.spatial.var = c(0, .1, .2, .3)

# now march through the different parameter combinations running the
simulations

for( i in 1:meta.sim.num) {
  start = Sys.time()
  simRepOut = simulationReplicator(Replications, sim.parameters[i, ], MC =
TRUE)
  simOut = simRepReorganizer(simRepOut)

  R2Output[as.character(sim.parameters[i, "sample.size"]),
           as.character(sim.parameters[i, "error.sd"]),
           as.character(sim.parameters[i, "B1.spatial.var"]),
           as.character(sim.parameters[i, "B2.spatial.var"]), , ] =
simOut[[1]]

  MetricOutput[as.character(sim.parameters[i, "sample.size"]),
               as.character(sim.parameters[i, "error.sd"]),
               as.character(sim.parameters[i, "B1.spatial.var"]),
               as.character(sim.parameters[i, "B2.spatial.var"]), , , ] =
simOut[[2]]
  end = Sys.time()

  print(paste("For loop", i,"of", meta.sim.num))
  print(round(difftime(end, start, units = "m"), 2))
  save(R2Output, MetricOutput, file =
"SpecificationSims/uberScriptOutput.RData")
}
```

I'm not going to run that code here (it took almost a month to run on the R Server), but let's load up the results and start to look at them. Or at least come up with some questions to ask of the data and a plan for the future.

```r
load("../Data/uberScriptOutput20120919.RData")
dimnames(MetricOutput)
```

```
## $ss
## [1] "50"   "100"  "200"  "500"  "1000"
##
## $error.sd
## [1] "2" "4" "6"
##
## $B1sv
## [1] "0"   "0.1" "0.2" "0.3"
##
## $B2sv
## [1] "0"   "0.1" "0.2" "0.3"
##
## $simNum
##   [1] "1"   "2"   "3"   "4"   "5"   "6"   "7"   "8"   "9"   "10"  "11"
##  [12] "12"  "13"  "14"  "15"  "16"  "17"  "18"  "19"  "20"  "21"  "22"
##  [23] "23"  "24"  "25"  "26"  "27"  "28"  "29"  "30"  "31"  "32"  "33"
##  [34] "34"  "35"  "36"  "37"  "38"  "39"  "40"  "41"  "42"  "43"  "44"
##  [45] "45"  "46"  "47"  "48"  "49"  "50"  "51"  "52"  "53"  "54"  "55"
##  [56] "56"  "57"  "58"  "59"  "60"  "61"  "62"  "63"  "64"  "65"  "66"
##  [67] "67"  "68"  "69"  "70"  "71"  "72"  "73"  "74"  "75"  "76"  "77"
##  [78] "78"  "79"  "80"  "81"  "82"  "83"  "84"  "85"  "86"  "87"  "88"
##  [89] "89"  "90"  "91"  "92"  "93"  "94"  "95"  "96"  "97"  "98"  "99"
## [100] "100"
##
## $optimized
##  [1] "AICc"     "corB0"    "corB1"    "corB2"    "CV"       "GCV"
##  [7] "R2"       "RMSE.B0"  "RMSE.B1"  "RMSE.B2"  "SCV"      "ttest%B0"
## [13] "ttest%B1" "ttest%B2"
##
## $metric
##  [1] "bandwidths" "B0.cor"     "B1.cor"     "B2.cor"     "B0.RMSE"
##  [6] "B1.RMSE"    "B2.RMSE"    "B0.t.perc"  "B1.t.perc"  "B2.t.perc"
## [11] "GCV"        "SCV"        "CV"         "AICc"       "R2"
##
```

```r
dimnames(R2Output)
```

```
## $ss
## [1] "50"   "100"  "200"  "500"  "1000"
##
## $error.sd
## [1] "2" "4" "6"
##
## $B1sv
## [1] "0"   "0.1" "0.2" "0.3"
##
## $B2sv
## [1] "0"   "0.1" "0.2" "0.3"
##
```

```
## $simNum
##   [1] "1"   "2"   "3"   "4"   "5"   "6"   "7"   "8"   "9"   "10"  "11"
##  [12] "12"  "13"  "14"  "15"  "16"  "17"  "18"  "19"  "20"  "21"  "22"
##  [23] "23"  "24"  "25"  "26"  "27"  "28"  "29"  "30"  "31"  "32"  "33"
##  [34] "34"  "35"  "36"  "37"  "38"  "39"  "40"  "41"  "42"  "43"  "44"
##  [45] "45"  "46"  "47"  "48"  "49"  "50"  "51"  "52"  "53"  "54"  "55"
##  [56] "56"  "57"  "58"  "59"  "60"  "61"  "62"  "63"  "64"  "65"  "66"
##  [67] "67"  "68"  "69"  "70"  "71"  "72"  "73"  "74"  "75"  "76"  "77"
##  [78] "78"  "79"  "80"  "81"  "82"  "83"  "84"  "85"  "86"  "87"  "88"
##  [89] "89"  "90"  "91"  "92"  "93"  "94"  "95"  "96"  "97"  "98"  "99"
## [100] "100"
##
## $R2
## [1] "OLS" "LWR"
##
```

So, we ran some simulations, varying the sample size of the data set, the standard deviation of the error term in the model and the degree of spatial variation in the model coefficients.

Each simulation was conducted as follows:

1. Grab the simulation parameters.
2. Generate the data according to the model and parameters.
3. Choose a number of observations to include in the Locally Weighted Regression.
4. Run Locally Weighted Regression on the data using the chosen bandwidth for each observation within the dataset.
5. Calculate a number of model metrics for each bandwidth
6. Repeat previous two steps for a number of bandwidths, ranging from only 5 data points to a model approaching a global Ordinary Least Squares model (in our case, we still had declining weights based on distance, but all observations received positive weight in the regresssion).
7. Collect data on each metric when each metric is optimized. For instance, when we choose the bandwidth associated with the lowest GCV score, what are the other metric values (β RMSEs, etc.)

We kept track of the following model performance metrics, the pseudo R^2 of the model results, the correlation between the $\hat{\beta}$ and the true β, the percent of the observations for which we can reject the null hypothesis that $\hat{\beta} = \beta$, cross validation scores (leave one out, generalized, and standardized according to Paez), lastly the AIC score.

0.1 Data Generation Process

The Data Generation Process is achieved using the `DataGen` function, the code for which is given below.

```
source("../SimFunctions.R")
DataGen
```

```
## function (sample.size, error.sd, B1.spatial.var, B2.spatial.var)
## {
##     n = sample.size
##     east = runif(sample.size) * 10
```

```
##     north = runif(sample.size) * 10
##     indep.var1 = runif(sample.size) * 10
##     indep.var2 = runif(sample.size) * 10
##     trueB0 = 0
##     trueB1 = B1.spatial.var * north + 1 - 5 * B1.spatial.var
##     trueB2 = B2.spatial.var * east + 1 - 5 * B2.spatial.var
##     error = rnorm(sample.size, 0, error.sd)
##     dep.var = trueB0 + indep.var1 * trueB1 + indep.var2 * trueB2 +
##         error
##     output = data.frame(dep.var, north, east, indep.var1, indep.var2,
##         trueB0, trueB1, trueB2, error)
##     output
## }
```

The dependent variable is produced as follows:

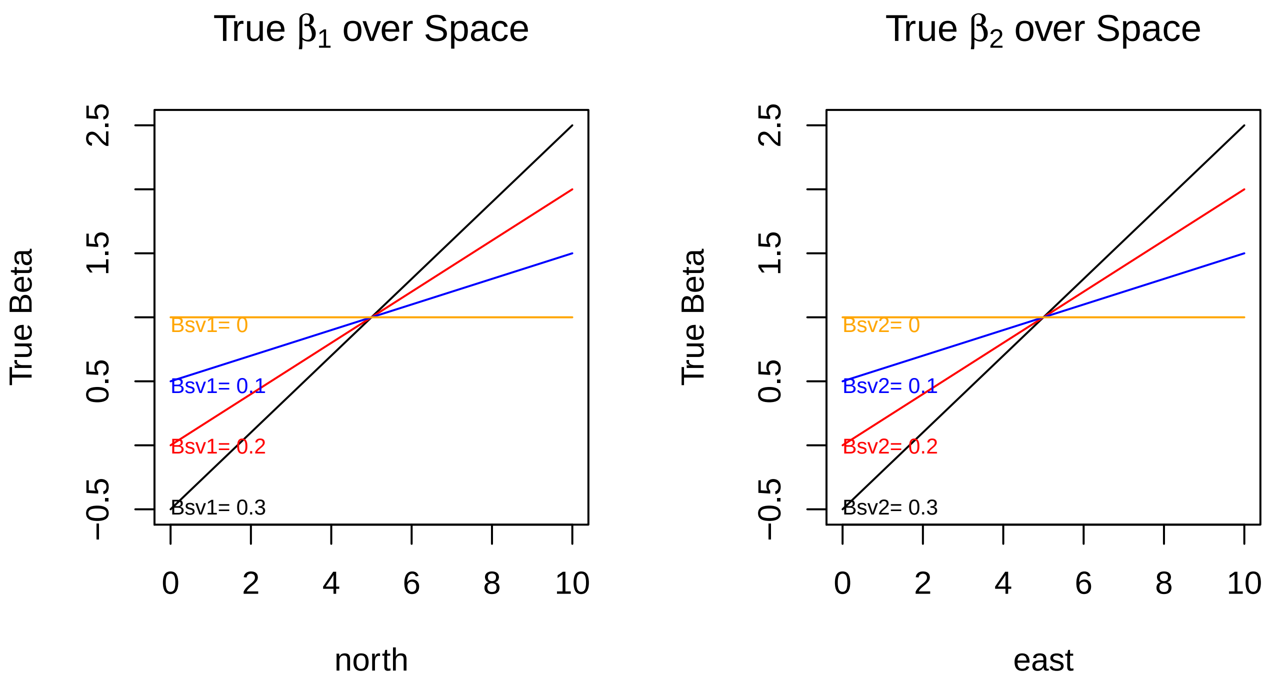
$$Y = \beta_0 + \beta_1(location)X_1 + \beta_2(location)X_2 + error \tag{1}$$

where $error \sim n(0, \sigma^2)$. Each observation is located within a geographic coordinate system $(east, north)$ where both $east$ and $north$ values are $\sim u(0, 10)$. The functions determining β_1 and β_2 are :

$$\beta_1(east, north) = 1 + Bsv1 * north - 5 * Bsv1 \tag{2}$$
$$\beta_2(east, north) = 1 + Bsv2 * east - 5 * Bsv2 \tag{3}$$

In our simulations we let Bsv_i vary $\{0, 0.1, 0.2, 0.3\}$, thus the relationship between βs and location can be visualized as:

```
east = north = 0:10
BetaFunc = function(x, Bsv) {
    1 + Bsv * x - 5 * Bsv
}
par(mfrow = c(1, 2))
plot(north, BetaFunc(north, 0.3), type = "l", xlab = "north", ylab = "True
Beta",
    main = expression(paste("True ", beta[1], " over Space")))
lines(north, BetaFunc(north, 0.2), col = "red")
lines(north, BetaFunc(north, 0.1), col = "blue")
lines(north, BetaFunc(north, 0), col = "orange")
text(rep(0, 4), seq(0.925, -0.5, length = 4), paste("Bsv1=", (0:3)/10),
    pos = 4, col = c("orange", "blue", "red", "black"), cex = 0.7, offset =
0)

plot(east, BetaFunc(east, 0.3), type = "l", xlab = "east", ylab = "True
Beta",
    main = expression(paste("True ", beta[2], " over Space")))
lines(east, BetaFunc(east, 0.2), col = "red")
lines(east, BetaFunc(east, 0.1), col = "blue")
lines(east, BetaFunc(east, 0), col = "orange")
text(rep(0, 4), seq(0.925, -0.5, length = 4), paste("Bsv2=", (0:3)/10),
    pos = 4, col = c("orange", "blue", "red", "black"), cex = 0.7, offset =
0)
```



Our simulations include data generation processes in which:

1. neither coefficient varies over space ($Bsv_1 = 0$ & $Bsv_2 = 0$)
2. both coefficients vary over space ($Bsv_1 \neq 0$ & $Bsv_2 \neq 0$)
3. only one coefficient varies over space ($Bsv_1 = 0$ & $Bsv_2 \neq 0$ OR $Bsv_1 \neq 0$ & $Bsv_2 = 0$)

Each simulation can be characterized by our selection of four data generation parameters,

- sample size $\{50, 200, 500, 1000\}$
- variance of the error term $\{2^2, 4^2, 6^2\}$
- degree of spatial variation in β_1 $\{0, .1, .2, .3\}$
- degree of spatial variation in β_2 $\{0, .1, .2, .3\}$

1 Research Questions

What do we want to know about LWR?

1. Are there systematic differences in the bandwidth size selected by different techniques? How do LOOCV, Standardaized CV, Generalize CV, and the AICc compare?
2. What sort of spatial variation in the coefficients is necessary relative to the error to need LWR?
3. If there is no spatial relationship, will LWR default back to global OLS?

2 Applying Locally Weighted Regression

After generating the data, we applied Locally Weighted Regression and calculated numerous diagnostics in order to measure the performance of the regression technique.

Locally Weighted Regression (LWR) is an estimation strategy allowing non-stationary model parameters. A vector of regression parameters is estimated using Equation (4) for each location within the dataset,

$$\hat{\beta}_{location_i} = (X^T W_{location_i} X)^{-1} X^T W_{location_i} Y, \tag{4}$$

where X is the standard n x m data matrix, Y the n x 1 vector of dependent variable values, and $W_{location_i}$ is an n x n weights matrix. We construct the weights matrix for a given location to give positive weights to the k-nearest data points, with weights declining according to a bi-square function as distances increase. Specifically, we create the weights matrix with zeros on the off-diagonal and calculate the jjth diagonal element as,

$$w_{ij} = \begin{cases} \left[1 - \left(\frac{d_{ij}}{d_{ik}}\right)^2\right]^2 & \text{if } d_{ij} \leq d_{ik} \\ 0 & \text{if } d_{ij} > d_{ik} \end{cases} \tag{5}$$

where d_{ij} is the distance between observations i and j, and d_{ik} is the distance to the kth nearest observation to observation i.

2.1 Cross-Validation

Theory does not provide guidance as to how many observations should receive positive weights in the local regression and must be determined by the researcher for the problem at hand. Typically, the k parameter is determined by minimizing a cross-validation metric. This research aims to systematically compare the performance of four different cross-validation metrics used in LWR research.

1. Leave-One-Out Cross-Validation
2. Generalized Cross-Validation
3. Standardized Cross-Validation
4. Akaike Information Criterion

Does choosing the optimal number of observations to include in the LWR through these four strategies yield similar results? If there are differences, are there patterns in how they are different?

2.2 Leave-One Out Cross-Validation

$$\sum (y - \hat{y}_{-i})^2 \tag{6}$$

2.3 Generalized Cross-Validation Score

$$n * \sum_{i=1}^{n} \frac{(y_i - \hat{y}_i)^2}{(n - v_1)^2}, \tag{7}$$

where y_i is the dependent variable value, $\hat{y}_i$ is the predicted dependent variable value for observation i, and v_1 is the "effective number of model parameters."[1] In an LWR model, the number of parameters

[1] v_1 =tr($\mathbf{S}$), where the matrix $\mathbf{S}$ is the "hat matrix" which maps y onto $\hat{y}$,

$$\hat{y} = \mathbf{S}y,$$

and each row of $\mathbf{S}$, r_i is given by:

$$r_i = X_i (X^T W(location_i) X)^{-1} X^T W(location_i).$$

to be estimated is no longer equal to the number of variables included because we allow the regression coefficients to vary over space. The GCV score calculates the "effective" number of model parameters, v_1, and penalizes the model for increasing the number of parameters without sufficient reduction in model accuracy. Taking the square root of Equation (7) and rearranging yields,

$$\sqrt{GCV} = \sqrt{\frac{n}{n - v_1}} \sqrt{\frac{\text{Sum of Squared Residuals}}{n - v_1}}, \tag{8}$$

which approaches $\hat{\sigma}$ as v_1 approaches m for large n. Henceforth, throughout the paper we report the square root of (7) because of its similarity to $\hat{\sigma}$.

2.4 Row Standardized Cross-Validation

Something about Paez, who wanted a CV score that was more robust to outliers.

$$\frac{\sum(y - y_{-i})^2}{\sum y} \tag{9}$$

2.5 Akaike Information Criterion

$$2 * n * ln(\hat{\sigma}) + n * ln(2 * \pi) + n * \frac{n + v_1}{n - 2 - v_1} \tag{10}$$

3 Which Bandwidths Do Selection Metrics Suggest?

In this section we compare the bandwidth selected by the different metrics.

3.1 Overall

- What are some summary stats about the bandwidths selected by each metric? (table: row for each metric, column for min, median, mean, max, sd)
- What is the visual distribution of bandwidths selected by each metric? (small multiples of a histogram for each metric)

```
mymetrics = c("CV", "GCV", "SCV", "AICc")
summary.table = matrix(0, length(mymetrics), 7)
for (mymetric in mymetrics) {
    summary.table[which(mymetrics == mymetric), 1:6] =
summary(MetricOutput[,
        , , , , mymetric, "bandwidths"])
    summary.table[which(mymetrics == mymetric), 7] = sd(MetricOutput[, , ,
,
        , mymetric, "bandwidths"])
}
rownames(summary.table) = mymetrics
colnames(summary.table) = c("min", "Q1", "median", "mean", "Q3",
    "max", "sd")
print(round(summary.table))
```

```
##       min Q1 median mean  Q3 max  sd
## CV     10 40     70  112 135 999 134
## GCV     5 40     70  111 135 999 134
## SCV    10 45     90  162 235 955 149
## AICc   15 45     80  121 145 999 134
```

Comparing the bandwidths at this level of aggregation is of limited use because we do not expect the same bandwidth to always be suggested. First, the bandwidth suggestion is constrained to be smaller than the sample size of the data, and so we should break out the simulation. Second, we expect the bandwidth selected to be a function of the degree of spatial variation in the underlying data generation process.

3.2 Sample Size

```
myss = c("50", "100", "200", "500", "1000")
mymetrics = c("CV", "GCV", "SCV", "AICc")

for (ssi in myss) {
    summary.table = matrix(0, length(mymetrics), 7)
    for (mymetric in mymetrics) {
        summary.table[which(mymetrics == mymetric), 1:6] =
summary(MetricOutput[ssi,
            , , , , mymetric, "bandwidths"])
        summary.table[which(mymetrics == mymetric), 7] =
sd(MetricOutput[ssi,
            , , , , mymetric, "bandwidths"])
    }
    rownames(summary.table) = mymetrics
    colnames(summary.table) = c("min", "Q1", "median", "mean", "Q3", "max",
        "sd")
    print(paste("Sample Size =", ssi))
    print(round(summary.table))
}
```

```
## [1] "Sample Size = 50"
##       min Q1 median mean Q3 max sd
## CV     10 20     30   31 40  49 12
## GCV     5 20     30   30 40  49 12
## SCV    10 25     30   29 35  49  8
## AICc   15 30     35   37 45  49  9
## [1] "Sample Size = 100"
##       min Q1 median mean Q3 max sd
## CV     10 30     45   50 65  99 24
## GCV     5 30     45   49 65  99 24
## SCV    15 45     50   51 60  99 13
## AICc   20 40     55   58 71  99 22
## [1] "Sample Size = 200"
##       min Q1 median mean  Q3 max sd
## CV     20 50     70   81 100 199 46
## GCV    15 45     70   80 100 199 46
## SCV    40 80     90   93 105 199 21
```

```
## AICc   30 55      80   90 110 199 44
## [1] "Sample Size = 500"
##       min  Q1 median mean  Q3 max  sd
## CV     30  85    120  151 170 499 107
## GCV    35  85    120  151 170 499 107
## SCV   100 190    215  217 235 480  38
## AICc   45  95    130  161 180 499 105
## [1] "Sample Size = 1000"
##       min  Q1 median mean  Q3 max  sd
## CV     55 130    185  246 260 999 210
## GCV    45 130    185  246 260 999 210
## SCV   255 380    420  421 455 955  62
## AICc   65 140    195  257 270 999 209
```

```
require(beanplot)
require(RColorBrewer)
mypal = brewer.pal(4, "Set2")

par(mfrow = c(3, 2))
par(oma = c(0, 0, 2, 0))
par(mar = c(2, 4.5, 3, 0))
myss = c("50", "100", "200", "500", "1000")
bws = c(2.5, 2.5, 5, 5, 10)
myssi = "1000"
for (myssi in myss) {
  beanplot(MetricOutput[ myssi, , , , , mymetrics[1], "bandwidths"],
           MetricOutput[ myssi, , , , , mymetrics[2], "bandwidths"],
           MetricOutput[ myssi, , , , , mymetrics[3], "bandwidths"],
           MetricOutput[ myssi, , , , , mymetrics[4], "bandwidths"],
           what = c(0, 1, 1, 0) ,
           log = "",
           bw = bws[which(myss == myssi)],
           cutmin = 5,
           cutmax = as.numeric(myssi),
           ylim = c(0, as.numeric(myssi)),
           xlim = c(0.5, 4.5),
           names = FALSE,
           main = paste("Sample Size = ", myssi),
           ylab = "",
           col = list(col = mypal[1], col = mypal[2], col = mypal[3], col =
mypal[4]),
           axes = FALSE)
  mtext(mymetrics, 1, line = 0, at = 1:4, col= mypal, font = 2)
  mtext("\\# of obs in LWR bandwidth", 2, line = 3, cex = .8)
  axis(2, las = 1)
  mtext("Bandwidth Distributions by Metric and Sample Sizes", 3,
        outer = TRUE, line = 0, cex = 1.5)

}
```

Notice that the distributions of selected bandwidths are similar for the CV, GCV, and AICc metrics,

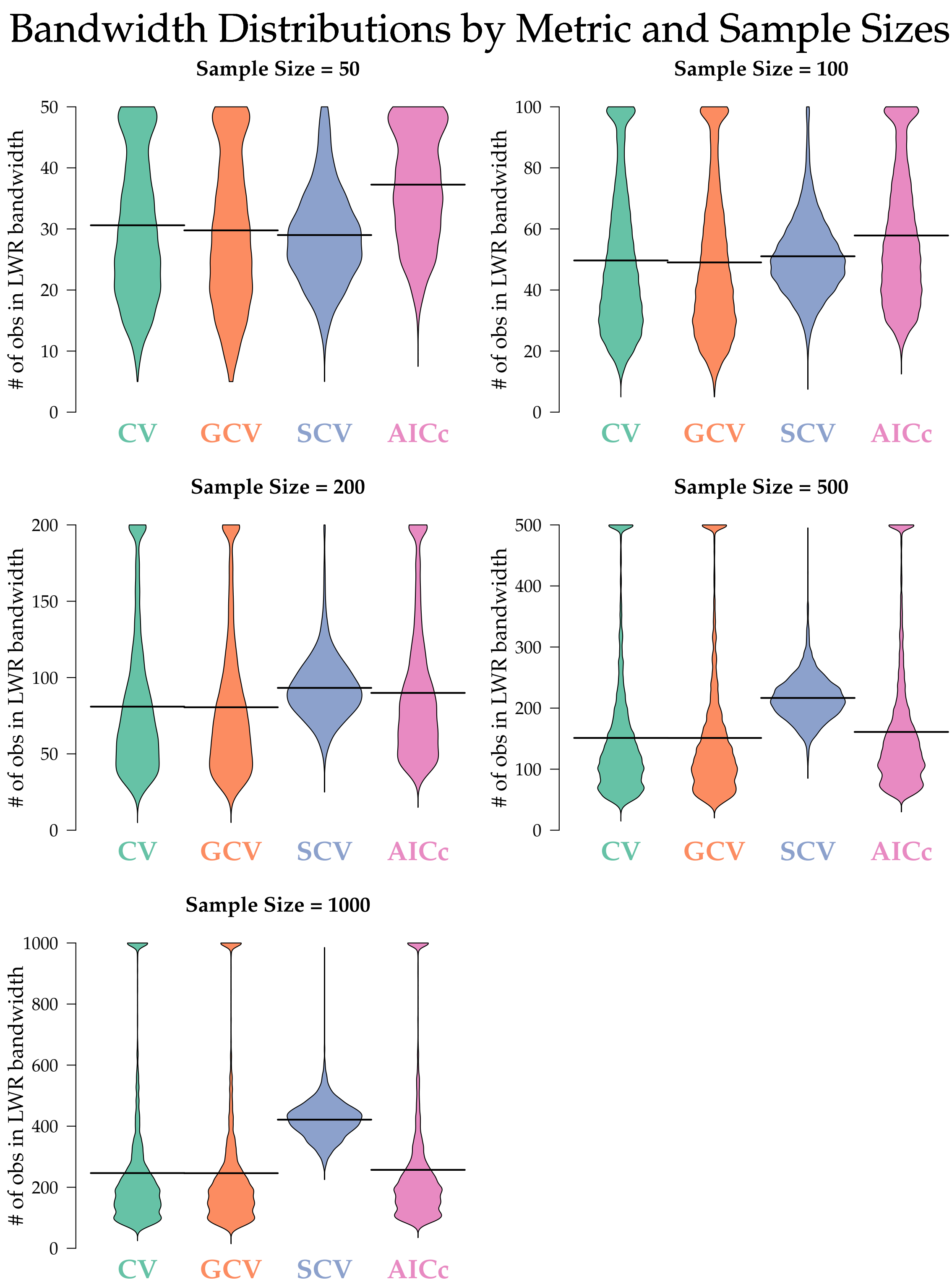


Figure 1: Hello World. This is a figure caption. I am going to keep writing for a bit to see what happens if I just keep writing and writing. What is the point of this graphic? You will find that out right here.

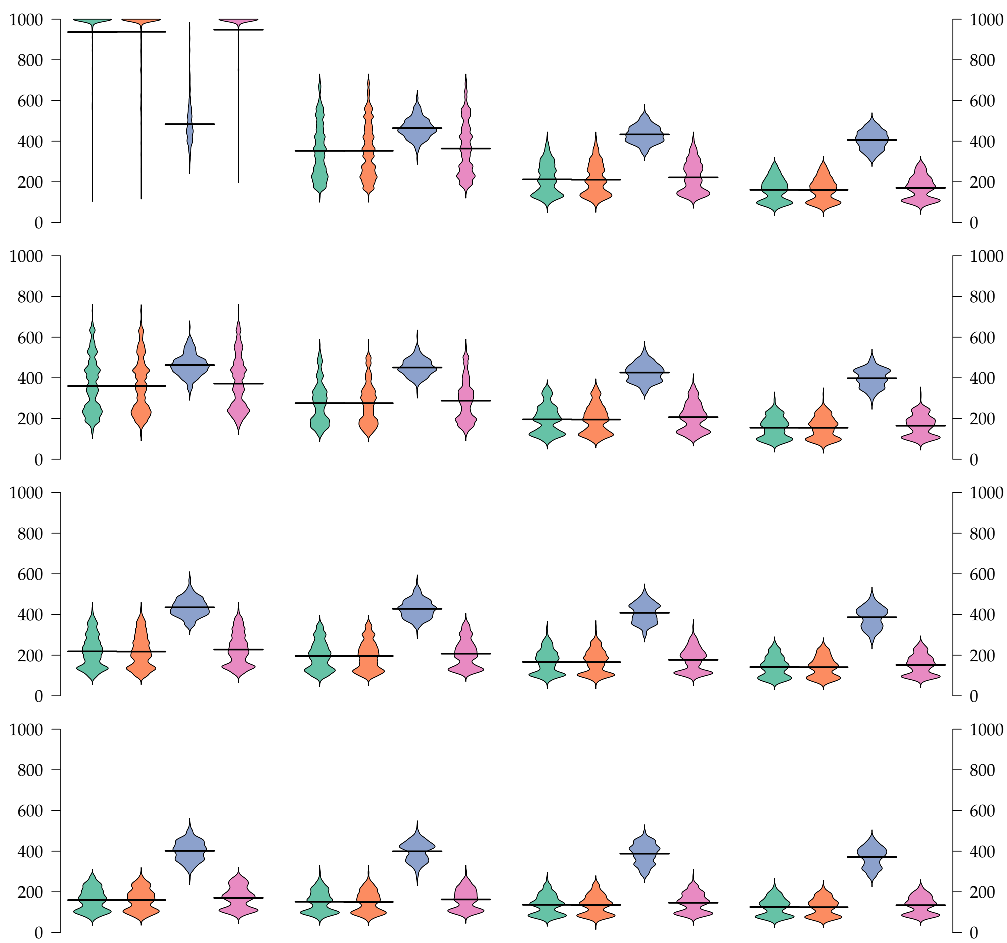
while the SCV metric distribution stands out, especially at higher bandwidths. Additionally, note that most distributions have a cluster of selected bandwidths near the sample size. Given that one simulation parameterization included no spatial variation within the data generation coefficients, it makes sense to see a cluster of large bandwidths, a model specification that approaches Ordinary Least Squares. We now proceed to show the distributions by degree of spatial variation in the model coefficients. Figure 1.

3.3 By Degree of Coefficient Variation

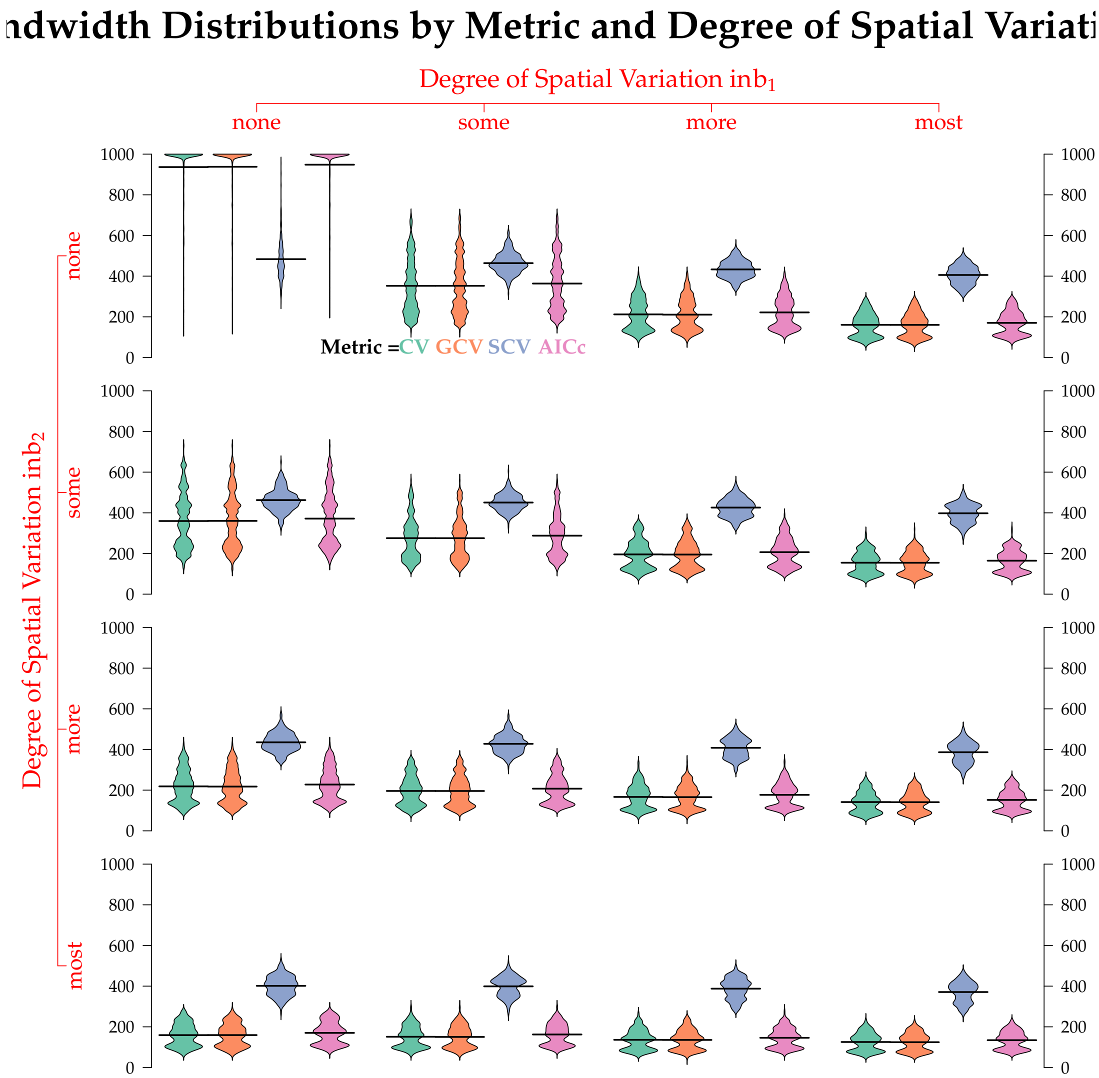
Challenge: We used four different levels of spatial variation in each of our two model coefficients, giving us a total of 16 spatial variation cases.

```
# The goal of this code is to make a 4 x 4 gird of beanplots showing the
distributions of optimal bandwidths across the four different LWR metrics
and the combinations of Bsv parameters.

require(RColorBrewer)
require(beanplot)
mypal = brewer.pal(4, "Set2")

# set some figure margin parameters

my.B1 = my.B2 = 1

df = layout( matrix(c(0, rep(17, 4),
                      18, 1:4,
                      18, 5:8,
                      18, 9:12,
                      18, 13:16), 5, 5, byrow = T),
             widths = c(.6, rep(1, 4)),
             heights = c(.6, rep(1, 4)))

#layout.show(df)

myssi = "1000"
myss = c("50", "100", "200", "500", "1000")
mymetrics = c("CV", "GCV", "SCV", "AICc")
bws = c(2.5, 2.5, 5, 5, 10)

par(oma = c(0, 0, 0, 2.5))
par(mar = c(.5, .5, .5, .5))

for (my.B2 in 1:4){ # four because there are four different B2sv parameter
files
  for (my.B1 in 1:4) { # four because there are four different B1sv
parameter files
    # now make a beanplot for the given Bsv1 and Bsv2
    beanplot(MetricOutput[myssi, , my.B1, my.B2, , mymetrics[1],
"bandwidths"],
             MetricOutput[myssi, , my.B1, my.B2, , mymetrics[2],
"bandwidths"],
             MetricOutput[myssi, , my.B1, my.B2, , mymetrics[3],
"bandwidths"],
```

```
                MetricOutput[myssi, , my.B1, my.B2, , mymetrics[4],
"bandwidths"],
                what = c(0, 1, 1, 0) ,
                log = "",
                bw = bws[which(myss == myssi)],
                cutmin = 5,
                cutmax = as.numeric(myssi),
                ylim = c(0, as.numeric(myssi)),
                #names = mymetrics,
                axes = FALSE,
                main = "",
                ylab = "",
                col = list(col = mypal[1], col = mypal[2], col = mypal[3], col
= mypal[4]))
      if(my.B1 == 1) axis(2, las = 1)
      if(my.B1 == 4) axis(4, las = 1)
  }
}
```



```
# Now work on the column labels
par(mar = c(0, 0, 0, 0))
spots = seq(.125, .875, l = 4)
plot(1, xaxs="i", xlim = c(0, 1), yaxs = "i", ylim = c(0, 1), type = "n",
axes = F)
axis(1, line = -2, at = spots,
     labels = F, col = "red")
text(spots, rep(0, 4), c("none", "some", "more", "most"),
     col = "red", pos = 3, cex = 1.3)
text(.5, .4, expression(paste("Degree of Spatial Variation in ", beta[1])),
     col = "red", cex = 1.5, font = 2)
#points((0:100)/100, rep(0, 101), col = c("red", rep("black", 9)))

# Now work on the row labels
plot(1, xaxs="i", xlim = c(0, 1), yaxs = "i", ylim = c(0, 1), type = "n",
axes = F)
axis(4, line = -5, at = spots,
     labels = F, col = "red")
text(rep(.5, 4), spots, c("most", "more", "some", "none"),
     col = "red", cex = 1.3, srt = 90)
text(.2, .5, expression(paste("Degree of Spatial Variation in ", beta[2])),
     col = "red", cex = 1.5, srt = 90)

# Title and Legend
mtext(paste("Bandwidth Distributions by Metric and Degree of Spatial
Variation"),
      outer = TRUE, line = -2.5, side = 3, font = 2, cex = 1.5, at = .52)
mtext(c("Metric =", "CV", "GCV", "SCV", "AICc"), outer = TRUE, line = -21,
side = 3,
      cex = .8, at = c(.3, .375, .41, .46, .508), adj = 0,
      col = c("black", mypal), font = 2)
```



4 How Accurate Are the Coefficient Estimates?

Rather than just looking at the bandwidths selected, researchers are probably more interested in the accuracy of the model predictions, specifically with regard to the model coefficients. In particular, does LWR tend to overfit the data by choosing small bandwidths and spurious coefficients? In this section we compare the estimated model coefficients to the true model coefficients to better understand the reliability of the LWR procedure.

5 What Happens When the Model is Misspecified?

In previous sections we assumed that the model to be estimated using LWR was properly specified. That is, bothe variables (X_1 and X_2) are included and their coefficients are allowed to vary over space to reflect the true data generation process. This section relaxes the assumption of a perfectly specified model and omits one variable in the regression. Our new regression equation becomes:

$$y = \alpha(location) + \beta_1(location)X_1 + error \tag{11}$$

An important question to consider in these circumstances is, “What happens when the omitted variable had a spatially varying coefficient, but the included variable coefficients are stationary?” Does LWR choose a large bandwidth and reflect the stationarity of the included model parameters? Does LWR select a small bandwidth and estimate spatially varying intercept terms? If so, what are the impacts on our estimates of the stationary parameter?