Tyler Behm
CSCI 5753
Graduate Student Project

Goodness-of-Fit of Gaussian Process Regression Map of Wifi Signal Strength on Farrand Field

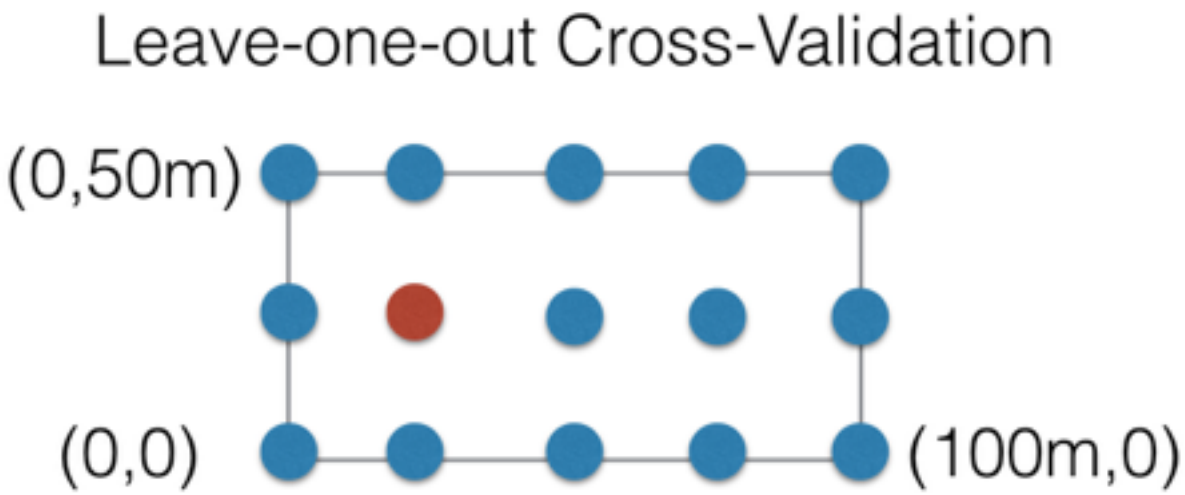
Introduction:

Kriging or Gaussian process regression uses Gaussian process governed by covariance to interpolate sample points and optimize the smoothness of fit. It was created by Danie G. Krige. He sought to map gold deposits in the Witwatersrand Reef in South Africa. He wanted to keep drilling for deposit samples to a minimum to prevent wasted expenses and damage. This is where the technique of kriging comes in handy. It quantifies the uncertainty between actual data samples in the interpolation map. Thus to decrease the uncertainty in the map as much as possible with only one more sample, simply sample at the highest uncertainty coordinate on the map.

Several papers have sought to use kriging to map wifi signal strength in complex or large environments like industrial facilities. Kriging provides a lower cost alternative to more expensive field testing services.

I suspect that Farrand Field gets its wifi coverage from surrounding buildings. Thus, I hypothesize that the signal strength map will be shaped like a bowl with the center of the field having the lowest signal strength.

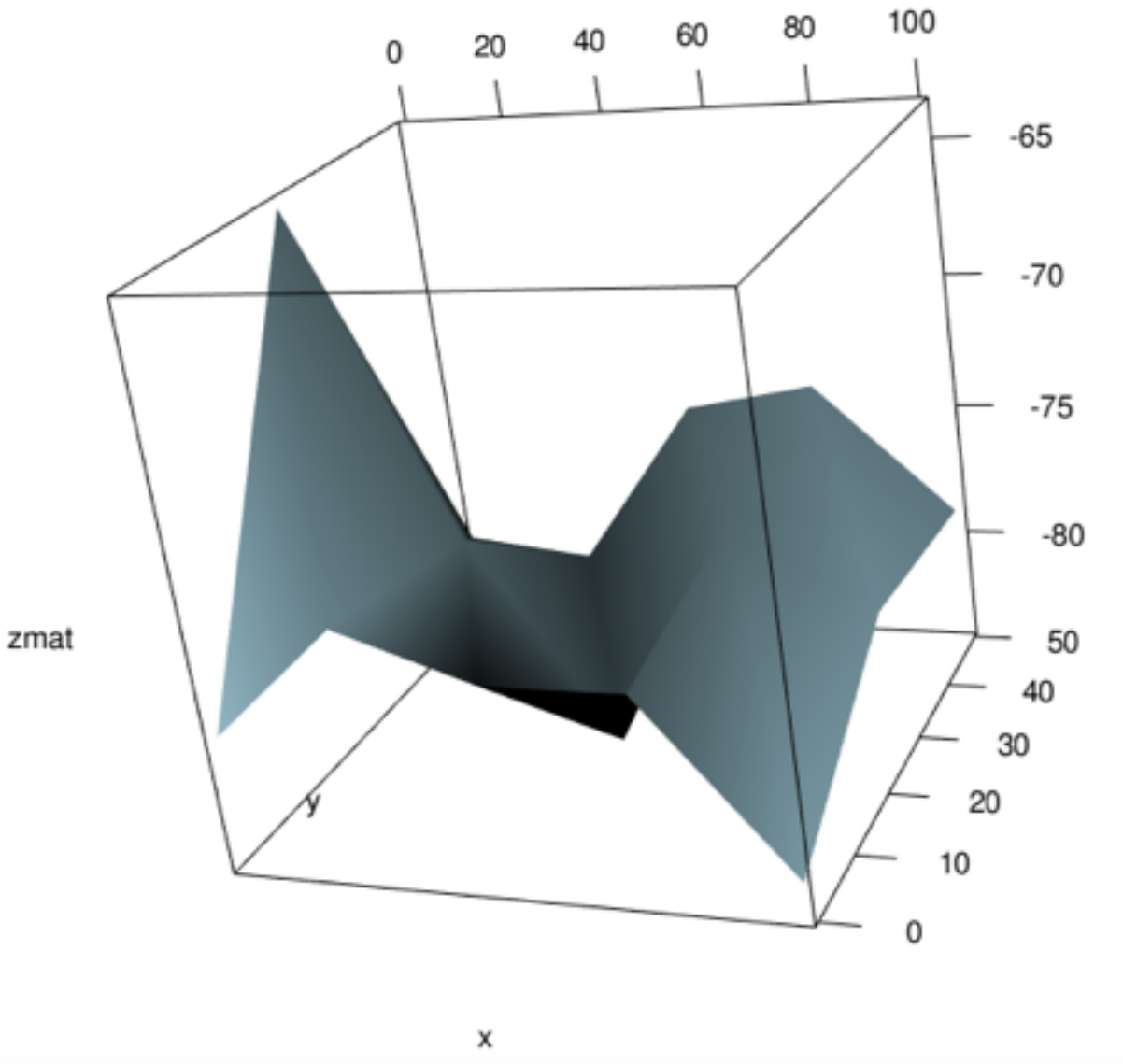
Although there were more gaussian process tools available to the python library, I chose to stick with R because I wanted to develop my R skills which are weaker than my python skills.



Collecting data:

For my project, I seek to use the Mac Airport tool to collect signal strength data and to use R to deduce an interpolation map using kriging. In the Joubert and Helberd paper, they sampled an area of about 100m by 10m at about 5m meter samplings. I have chosen to sample Farrand Field because it is about 100m by 50m. My attempts at using cell phone/laptop GPS have been unsuccessful, providing accuracy of only within 30m. Thus I intended to divide Farrand into a grid as seen below. The code to collect the samples is given in the appendix.

I collected 20 samples separated by 15 seconds at each of the 15 gridpoints. The data was collected on April 3rd, 2016 from the hours of 11am to 1pm. Below is the map of data with the values of wifi signal strength given along the z-axis with units of decibels.



Examining the Data:

Visually, my data seems reasonably distributed. There is weak wifi signal in the middle and strong wifi signal at the edges. This gives my map a quasi-bowl-shape as I hypothesized. This is because of the source of wifi coming from building near the edges of Farrand Field.

The strongest signal comes from the data point x = 0 m and y = 25 m with wifi signal value of z = -63.85 dB. The source of this wifi is from the Warner Imig Music building. Considering the proximity of the building to that data point, it makes sense that the signal is the strongest because there isn't much distance for the signal to attenuate over.

The second strongest signal comes from the data point x = 75 m and y = 50 m with wifi signal value of z = -74.15 dB. The source of this wifi is from the Baker Hall building. The signal value is significantly weaker than the value near Warner Imig Music building. This makes sense considering that the Baker Hall building is much further from Farrand Field than is Warner Imig Music building. This give the signal more distance over which to attenuate.

I roughly estimated that the Music building is at the grid point x = -25 m and y = 25 and that Baker Hall is at the grid point x = 60 m and y = 150 m. The Music building is about 50 m from its closest data point. Baker Hall is ((75-60)^2 + (50 - 150)^2)^.5 = 101 m from its closet data point. Thus the signal has twice as much distance to attenuate.

Building a Variogram Model:

When using the kriging tool in R, we are asked to input a variogram model. In R, it takes the form vgm(sill, type, nugget, range). We explain the general concept of a variogram below before detailing the variogram model that I derived.

Sill == The Maximum Semivariance

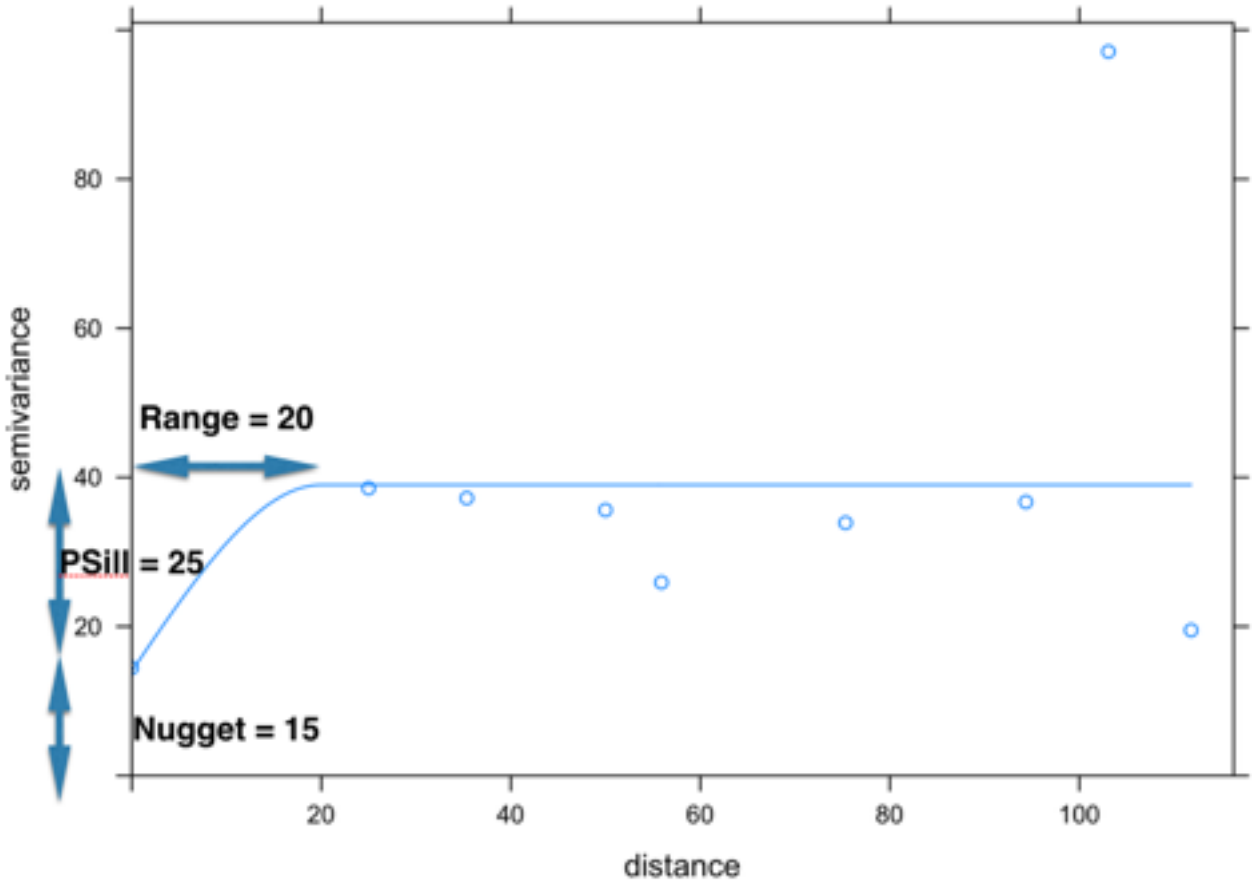
Wikipedia describes the sill as "Limit of the variogram tending to infinity lag distances." It's the maximum amount that data samples vary by when separated past decoupling. Usually, it is described in terms of partial sill which is the difference of the sill and the nugget.

Type == Spherical:

This states the type of the model that we want to put use in our variogram. This can come in many types including spherical, gaussian, and exponential. These models differ by how they rise after the nugget and how the sill grows at large values. The only way to know which model to use is to actually fit one to your data. I found that the spherical model fit the best for my data.

Nuggets == Standard Deviation of Sample Data:

Wikipedia describes the nugget as “The height of the jump of the semivariogram at the discontinuity at the origin.” This is just a simple matter of terminology cross over between geologist and everyone else. Colloquially, what is a nugget? It’s when you hit a particularly dense patch in a small sample. A chocolate chip is a nugget within a chocolate chip cookie. Consider that whole cookie to be below the spacial accuracy limit of sampling for our kriging but our measurement probe still has the spacial resolution beyond than of a chocolate chip. Then you have a some chance of hitting a chocolate chip nugget and some chance of hitting just cookie dough. Even though these two samples are at the same point, they provide two very different realizations of the chocolate density probability field at that cookie. And thus, nuggets are just analogous to what every other scientific field would call standard deviation at that sample point.



By setting nuggets to zero, we are telling R that the sample is 100% accurate. We are saying that we will measure the same wifi signal strength at that point regardless of the time of day, network traffic, weather, etc. But we know this is not the case. So we set the nugget equals to one to tell R that the signal strength is normally distributed.

Range == How Long Until Adjacent Samples are Decoupled

Wikipedia describes the range as follows: “The distance in which the difference of the variogram from the sill becomes negligible. In models with a fixed sill, it is the distance at which this is first reached; for models with an asymptotic sill, it is conventionally taken to be the distance when the semivariance first reaches 95% of the sill.”

Because my data fit a spherical model, I didn’t need to worry about an asymptotic sill. Thus the range is apparent without the 95% calculation.

My Variogram Model:

I have attached an image with my model and the accompanying components pointed out. Thus my variogram model is given by vgm(25, “Sph”, 20, 15). The code that I used to deduced this model is given in the appendix.

Leave-one-out Cross-Validation:

I will use the samples at the blue points to build my kriging map. Then I will use the reserved sample at the red points to quantify the quality of my map.

How to do Kriging in R:

The hardest part about doing the kriging is searching through tomes of R documentation to find the the right package, library, and function and then deciphering how to use it. Thankfully, I found a R function that does exactly what I want after much searching.

```
w <- krige.cv(z~1, df, df, model = m)
```

This function takes the linear model and data set to perform n-fold cross-validation interpolation onto the new coordinate system. Let me explain the exact details of how I have it setup. The first argument means that I’m interpolating on the signal strength. The second argument is the data set that I’m using for the interpolation scheme. The third argument is the new coordinate system. I chose to map it back on itself so that I can see if the interpolated values match the measured values. The fourth argument is the variogram model that I calculated in the previous section. Implicitly defined in this argument list are several default values. The most import default default value sets the number of folds such that the cross-validation is leave-one-out.

Leave-one-out Cross-Validation Results:

```
  var1.pred var1.var observed  residual      zscore fold   x  y
1  -77.07500 42.85714   -79.10 -2.025000 -0.30932386    1   0  0
2  -77.37500 42.85714   -74.90  2.475000  0.37806249    2  25  0
```

3	-77.25000	42.85714	-76.65	0.600000	0.09165151	3	50	0
4	-77.25000	42.85714	-76.65	0.600000	0.09165151	4	75	0
5	-76.80714	42.85714	-82.85	-6.042857	-0.92306168	5	100	0
6	-78.16429	42.85714	-63.85	14.314286	2.18654326	6	0	25
7	-77.27857	42.85714	-76.25	1.028571	0.15711688	7	25	25
8	-76.73214	42.85714	-83.90	-7.167857	-1.09490826	8	50	25
9	-77.34643	42.85714	-75.30	2.046429	0.31259713	9	75	25
10	-77.12500	42.85714	-78.40	-1.275000	-0.19475947	10	100	25
11	-76.96071	42.85714	-80.70	-3.739286	-0.57118533	11	0	50
12	-76.91429	42.85714	-81.35	-4.435714	-0.67756655	12	25	50
13	-77.36429	42.85714	-75.05	2.314286	0.35351298	13	50	50
14	-77.42857	42.85714	-74.15	3.278571	0.50081006	14	75	50
15	-77.07857	42.85714	-79.05	-1.971429	-0.30114069	15	100	50

Of my 300 data points, 226 which is 75% of them fit within one standard deviation of their predicted value. This is about what we would expect considering that a standard deviation should contain 68% of the data points. So my model is a good fit.

Conclusion:

Kriging is a good technique, but it is limited by the spatial resolution of my data. The variogram suggests that semi variance of two points are unrelated if those points are separated by more than 15 meters. To sample Farrand Field at 10 meter spacing would take 50 grid points. Using 20 data points per grid point with 15 second spacing would take more than 5 minutes * 50 = 4 hours and 10 minutes to record. At this far into my project, it's out of the scope of the project to commit that much more time plus the additional analysis.

If the data was resampled with 10 m resolution, the accuracy of the kriging model would increase greatly. The kriging model would be able to predict the large spike in signal at x = 25m and y = 50m because of the sample that are that with 10 meters of it and are thus coupled. This would significantly reduce the residual and tighten the confidence bounds at that point.

This was a great project to work on. I got to learn very much about the intersection of computer science and applied analytics. Thank you very much for the project idea.

Citations:

Duvallet, F., Tews, A. , WiFi Position Estimation in Industrial Environments Using Gaussian Processes

Joubert, PJ., Helberd, ASJ, An investigation into the use of the kriging for Wi-Fi RSSI estimation in complex indoor environments

Ferris, B., Hahnel, D., Fox, D., Gaussian Processes for Signal Strength-Based Location Estimation

Phillips, Caleb, Geostatistical Techniques for Practical Wireless Network Coverage Mapping

Wackernagel, Hans, Multivariate Geostatistics; an introduction with applications, 2nd edition

http://www.inside-r.org/packages/cran/gstat/docs/krige.cv

http://scikit-learn.org/stable/modules/gaussian_process.html

http://www.mactricksandtips.com/2010/06/use-terminal-to-measure-wi-fi-strength.html

https://www.msu.edu/~ashton/classes/866/notes/lect15/index.html

http://www.ncbi.nlm.nih.gov/pmc/articles/PMC4610440/

Code Appendix:

```
#! /bin/bash
#CODE TO COLLECT DATA
STR=~/Documents/cpm/project/data/x100y25.data
echo 'RSSI' > $STR
for i in `seq 1 20`; do /System/Library/PrivateFrameworks/Apple80211.framework/Versions/
Current/Resources/airport -I | grep CtlRSSI | awk '{print $2}' >> $STR; sleep 15.; done
```

```
#Variogram model calculations
library(sp)
library(gstat)
library(kriging)
library(maps)
library(rgl)

ftox <- function(f){
 words = strsplit(f, "y")
    x1 = words[[1]][1]
    x1 = strsplit(x1, "x")
    as.numeric(x1[[1]][2])
}

ftoy <- function(f){
 words = strsplit(f, "y")
    y1 = words[[1]][2]
    y1 = strsplit(y1, ".d")
    y = as.numeric(y1[[1]][1])
}

dist1 <- function(x, y){
    sqrt((x+25)^2 + (y-25)^2)
}
```

```
dist2 <- function(x, y){
    sqrt((x-60)^2 + (y-150)^2)
}

x=c()
y=c()
d1=c()
d2=c()
z=c()
files = c("x0y0.data", "x25y0.data", "x50y0.data", "x75y0.data",
 "x100y0.data",
"x0y25.data", "x25y25.data", "x50y25.data", "x75y25.data",
 "x100y25.data",
"x0y50.data", "x25y50.data", "x50y50.data", "x75y50.data",
 "x100y50.data")
for (f in files)
{
    dir  = paste("~/Documents/cpm/project/data/", f, sep = "")
    data = data.frame(read.csv(dir, sep=',', header=T))
 xf = ftox(f)
 yf = ftoy(f)
    for ( zeta in data$RSSI ){
        x  = append(x, xf)
        y  = append(y, yf)
        d1 = append(d1, dist1(xf, yf))
        d2 = append(d2, dist2(xf, yf))
        z  = append(z, zeta)
    }
}
df = data.frame(x, y, d1, d2, z)
coordinates(df) = ~x+y
#v = variogram(z~1, df, cutoff = 10000, width = 10)
#v.fit = fit.variogram(v, model = vgm(25, "Sph", 20, 14))
#plot(v, v.fit)

# Krige map calculations
library(sp)
library(gstat)
library(kriging)
library(maps)
library(rgl)

ftox <- function(f){
```

```
 words = strsplit(f, "y")
    x1 = words[[1]][1]
    x1 = strsplit(x1, "x")
    as.numeric(x1[[1]][2])
}

ftoy <- function(f){
 words = strsplit(f, "y")
    y1 = words[[1]][2]
    y1 = strsplit(y1, ".d")
    y = as.numeric(y1[[1]][1])
}

x=c()
y=c()
z=c()
files = c("x0y0.data", "x25y0.data", "x50y0.data", "x75y0.data",
 "x100y0.data",
    "x0y25.data", "x25y25.data", "x50y25.data", "x75y25.data",
 "x100y25.data",
    "x0y50.data", "x25y50.data", "x50y50.data", "x75y50.data",
 "x100y50.data")
for (f in files)
{
 dir = paste("~/Documents/cpm/project/data/", f, sep = "")
 data = data.frame(read.csv(dir, sep=',', header=T))
 xf = ftox(f)
 yf = ftoy(f)
 x  = append(x, xf)
 y  = append(y, yf)
    z  = append(z, mean(data$RSSI))
}
df = data.frame(x, y, z)
coordinates(df) = ~x+y
m <- vgm(25, "Sph", 20, 15)
w <- krige.cv(z~1, df, df, model = m) ## This is the proper way to
 make sure that the krige interpolation "bends" to fit the points
zmat <- matrix(df$z, 5,3)
x = c(0,25,50,75,100)
y = c(0, 25,50)
persp3d(x=x, y=y, z=zmat, col = "lightblue")
```