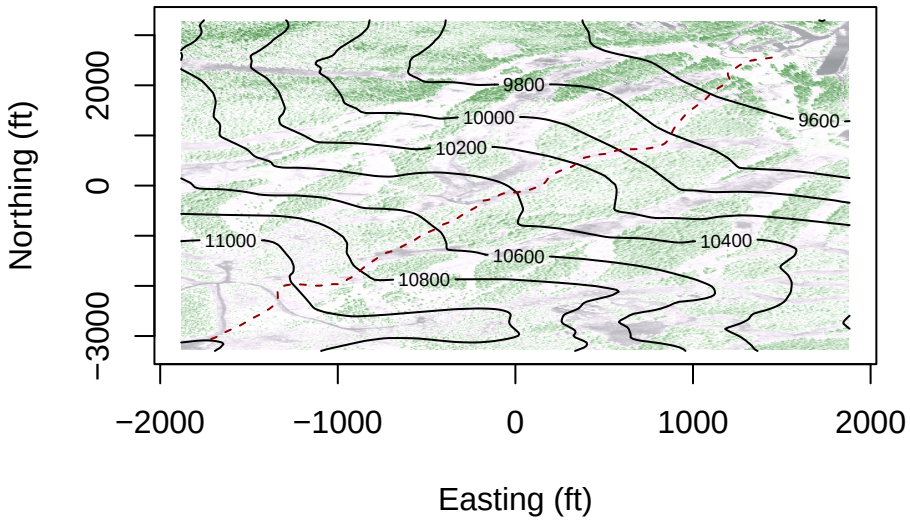
Spatial Experiments Doc 5

Antony Sikorski

Setup and Background:

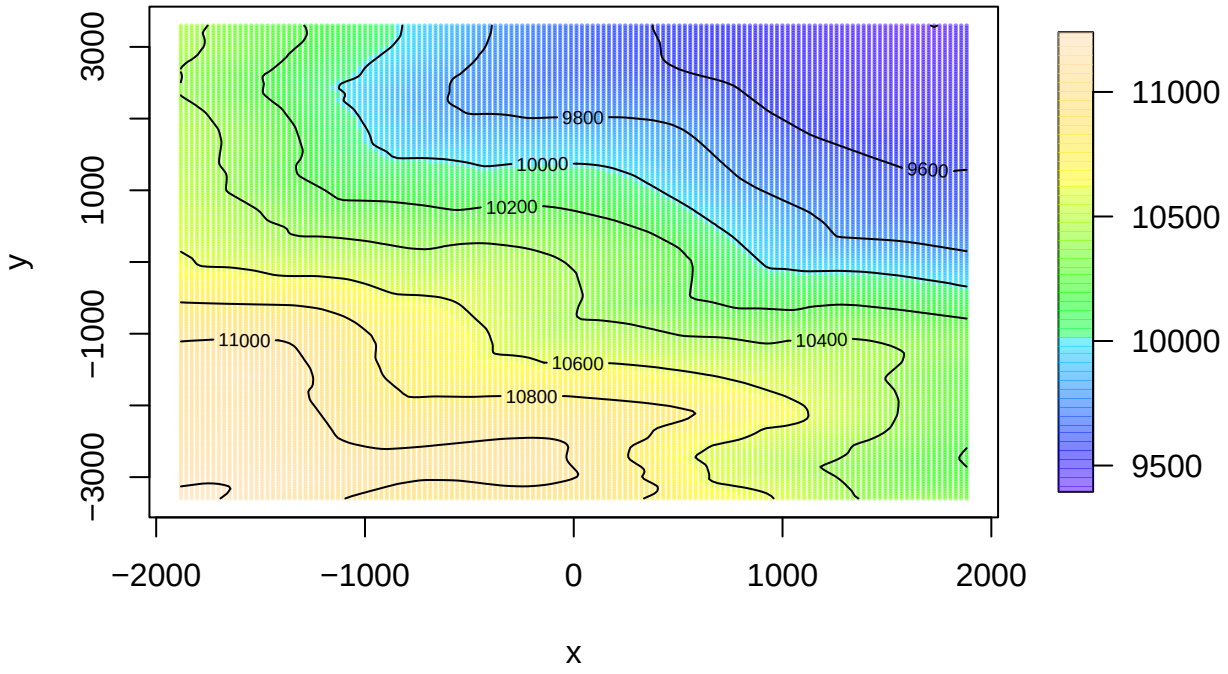
We are working with high resolution aerial images and elevations (all in units of feet) for part of the Mary Jane Ski Resort. This is stored as a large raster file, so we will have to do some conversions and wrangling. Also included are the locations of a ski trail down a particular part of the mountain. The trail locations do not fall onto the elevation grid locations – hence the need for spatial fitting. The top of the Super Gauge lift is at lower left corner of the image, and the Mary Jane base lodge is at the upper right. The ski trail that we will focus in on follows the Sluice Box run to Super Gauge and finishes on the Rifle Sight/Branch Line trails.

```
#libraries and data
suppressMessages(library( fields))
suppressMessages(library( raster))
suppressMessages(library(scales))
suppressMessages(library(ggplot2))
load("data.rda")
plot( range(RifleDEM$x),range(RifleDEM$y), type="n",
xlab="Easting (ft)", ylab="Northing (ft)")
image(rasterRifleImage, col=MJ.colors(256), add=TRUE)
contour(RifleDEM, add=TRUE)
lines( rifle.trail, col="red4", lty=2)
```



Converting to locations to use spatial fitting:

```
s<- make.surface.grid( RifleDEM[c("x","y")] )
z<- c( RifleDEM$z)
bubblePlot( s,z, col= alpha(topo.colors(256),.5),
highlight=FALSE, size=.3)
contour(RifleDEM, col="black", add=TRUE)
```



The code above provides good visual proof that our elevations have been properly matched to the locations. This is because the bubble plot values/colors match up to the original contour lines of the data, and the overall shape of the gradient is consistent with what we would expect from a mountain.

Interesting Lat Long Observation:

Let us look at the difference in width (in feet) when it comes to the top of our grid, and the bottom.

```
topLeft <- matrix(c(min(RifleDEM0$x),max(RifleDEM0$y)), nrow = 1)
topRight <- matrix(c(max(RifleDEM0$x),max(RifleDEM0$y)), nrow = 1)

topDist <- rdist.earth(topLeft, topRight)[1,1] * 5280
topDist
```

```
## [1] 3763.175
```

```
botLeft <- matrix(c(min(RifleDEM0$x),min(RifleDEM0$y)), nrow = 1)
botRight <- matrix(c(max(RifleDEM0$x),min(RifleDEM0$y)), nrow = 1)

botDist <- rdist.earth(botLeft, botRight)[1,1] * 5280
botDist
```

```
## [1] 3764.166
```

```
warp <- botDist - topDist
warp
```

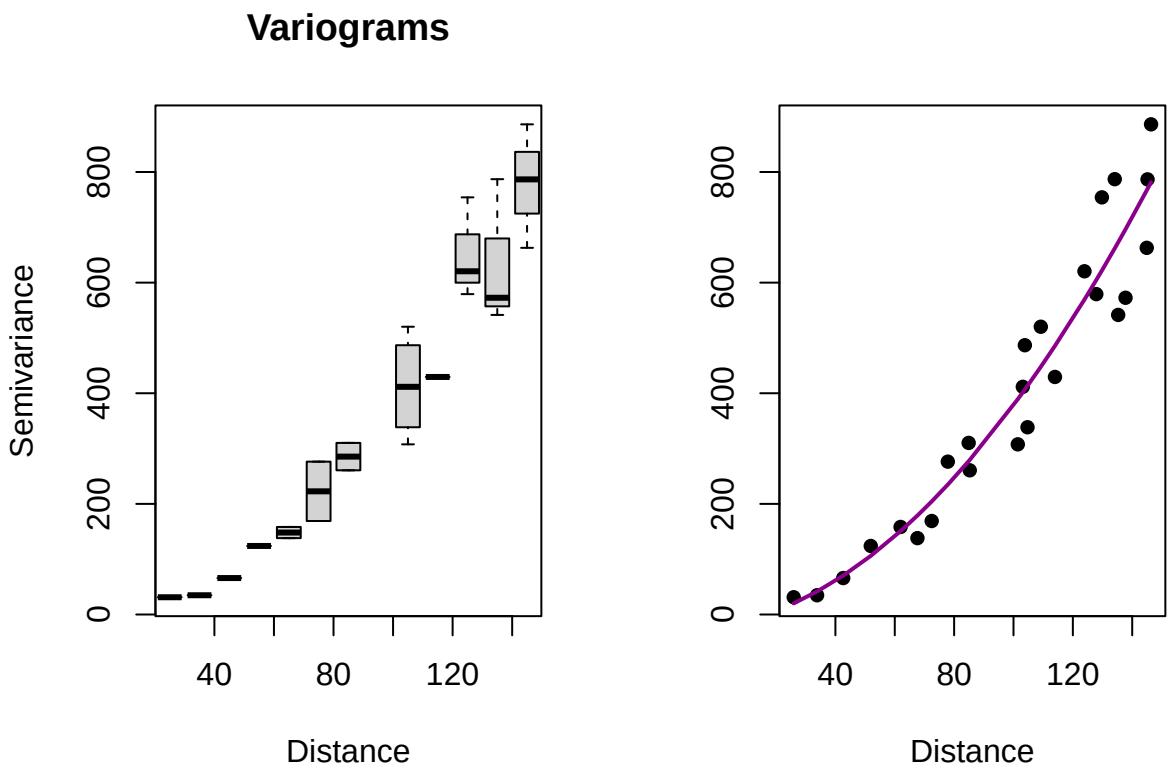
```
## [1] 0.9905512
```

The difference between the top (3763.175 ft) and the bottom (3764.166 ft) of the grid is roughly 1 foot, with the bottom of the grid being wider (since we are drawing closer to the equator). This minimal difference validates the fact that local maps have no need for adjusting for earth's curvature.

Variograms

Making a variogram for the data:

```
#average distance for x and y coords
dx<- mean(diff((RifleDEM$x)))
dy<- mean(diff((RifleDEM$y)))

#vgram matrix
look<- vgram.matrix(RifleDEM$z, R=150, dx=dx, dy=dy)

#plotting the variogram bar chart and points
par(mfrow = c(1,2))
bplot.xy( look$d, look$vgram, xlab = "Distance", main = "Variograms", ylab = "Semivariance")
plot(look$d, look$vgram, pch = 16, xlab = "Distance", ylab = "")

#fitting a quadratic model
fit <- lm(look$vgram ~ look$d + I((look$d)^2))

#use model to get predicted values
pred <- predict(fit)
ix <- sort(look$d, index.return = TRUE)$ix

#add polynomial curve to plot
lines(look$d[ix], pred[ix], col='darkmagenta', lwd = 2)
```




```
par(mfrow = c(1,1))

summary(fit)
```

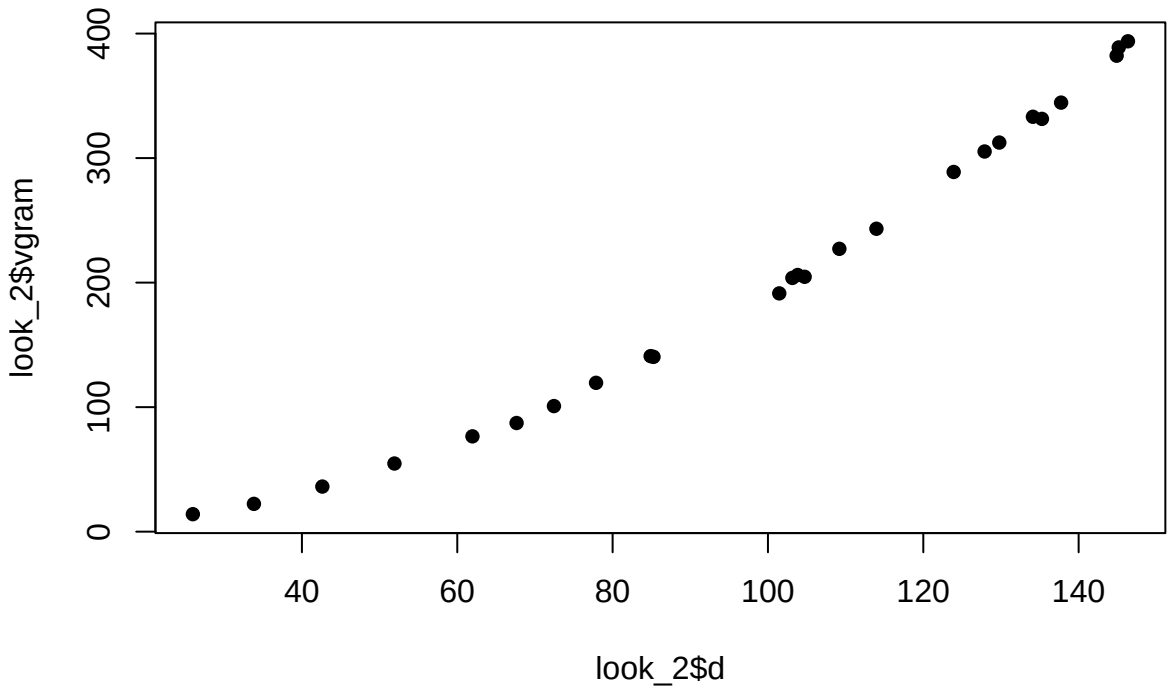
```
##
## Call:
## lm(formula = look$vgram ~ look$d + I((look$d)^2))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -131.245  -40.825    7.528   41.490  132.314
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -22.55458   96.56536  -0.234   0.8175
## look$d          0.81684    2.30273   0.355   0.7262
## I((look$d)^2)   0.03197    0.01247   2.563   0.0177 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 75.33 on 22 degrees of freedom
## Multiple R-squared:  0.9207, Adjusted R-squared:  0.9135
## F-statistic: 127.7 on 2 and 22 DF,  p-value: 7.821e-13
```

A few observations:

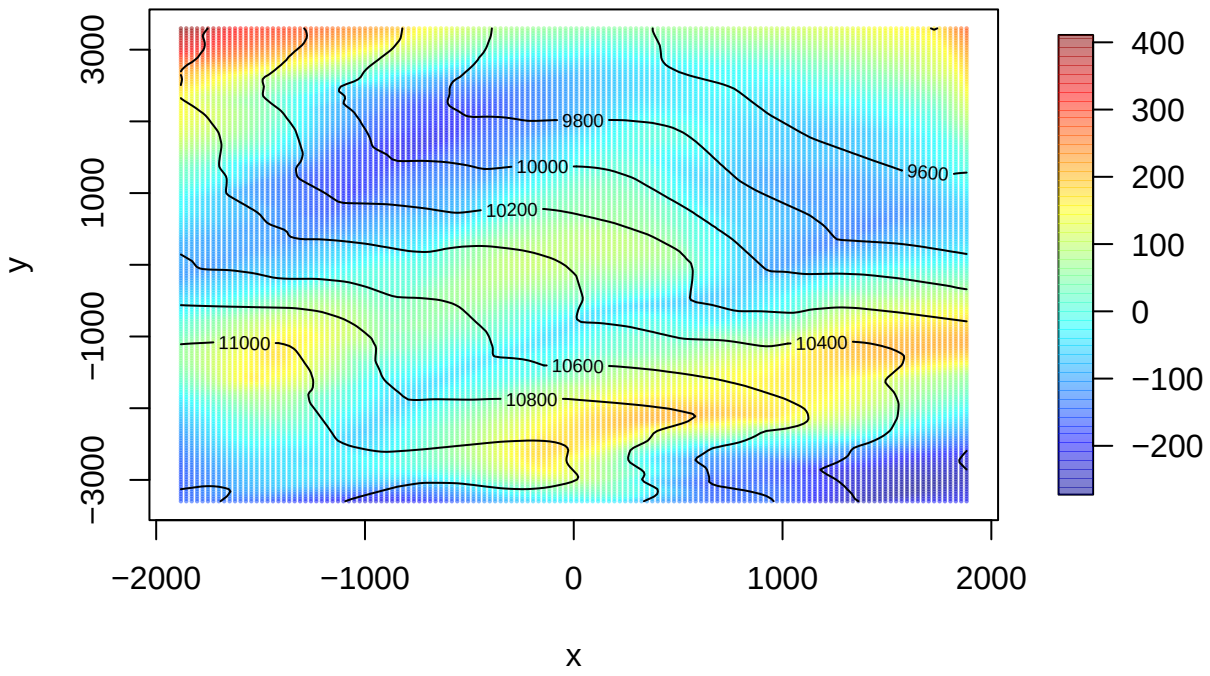
There is little evidence of large nugget variance. The intercept of the plot appears close to zero, at least relative to the sizes of the semivariance between the points. This plot does not look like it follows an exponential at small distances, but if the radius of distances is bumped up, we effectively tease out some behavior that looks similar to an exponential. Meanwhile, at small distances, it appears that a quadratic fit lines up nicely.

Now we plot a variogram and bubble plot for the residuals:

```
ZResidual<- lm(z ~ s)$residuals
nx<- nrow( RifleDEM$z)
ny<- ncol( RifleDEM$z)
RifleDEMResidual<- matrix(ZResidual,nx,ny )

#plotting variogram
RifleDEMResidual <- matrix(ZResidual,nx,ny )
look_2<- vgram.matrix(RifleDEMResidual, R=150, dx=dx, dy=dy)

plot(look_2$d, look_2$vgram, pch = 16)
```




```
s_1<- make.surface.grid( RifleDEM[c("x","y")] )
z_1<- c( RifleDEMResidual)

#residual plot
bubblePlot( s_1,z_1, col= alpha(tim.colors(256),.5),
highlight=FALSE, size=.3)
contour(RifleDEM, col="black", add=TRUE)
```



Observations:

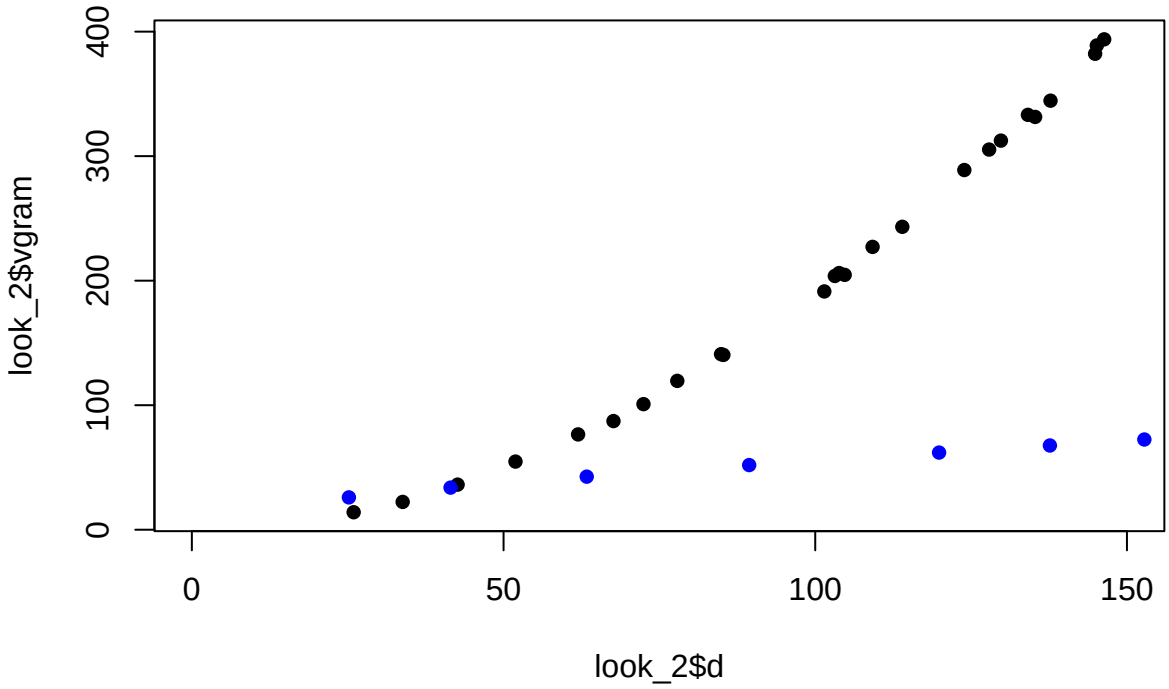
The shape of the variogram changes significantly. The ascent is much slower, and the maximum variance is significantly less at a distance of 150.

We can't fit a spatial surface to the whole data set, so first we increase the default storage.

```
RUN<- FALSE
if(RUN){
options(spam.nearestdistnnz=c(2E8,500))
system.time(
obj<- fastTps( s,z, aRange=300, lambda=0)
)
save(obj, file="myFit.rda")
}
load("myFit.rda")
```

Now we plot the variagram from this model (on top of the old variogram). The points for this one will be blue.

```
sigma2Hat<- obj$summary["sigma2"]
D <- look_2$d

cov <- Wendland(D, aRange=300, dimension=2, k=2)
cov_function <- sigma2Hat - sigma2Hat * cov
plot(look_2$d, look_2$vgram, pch = 16, xlim=c( 0,150))
points(cov_function, D, col = "blue", pch = 16)
```

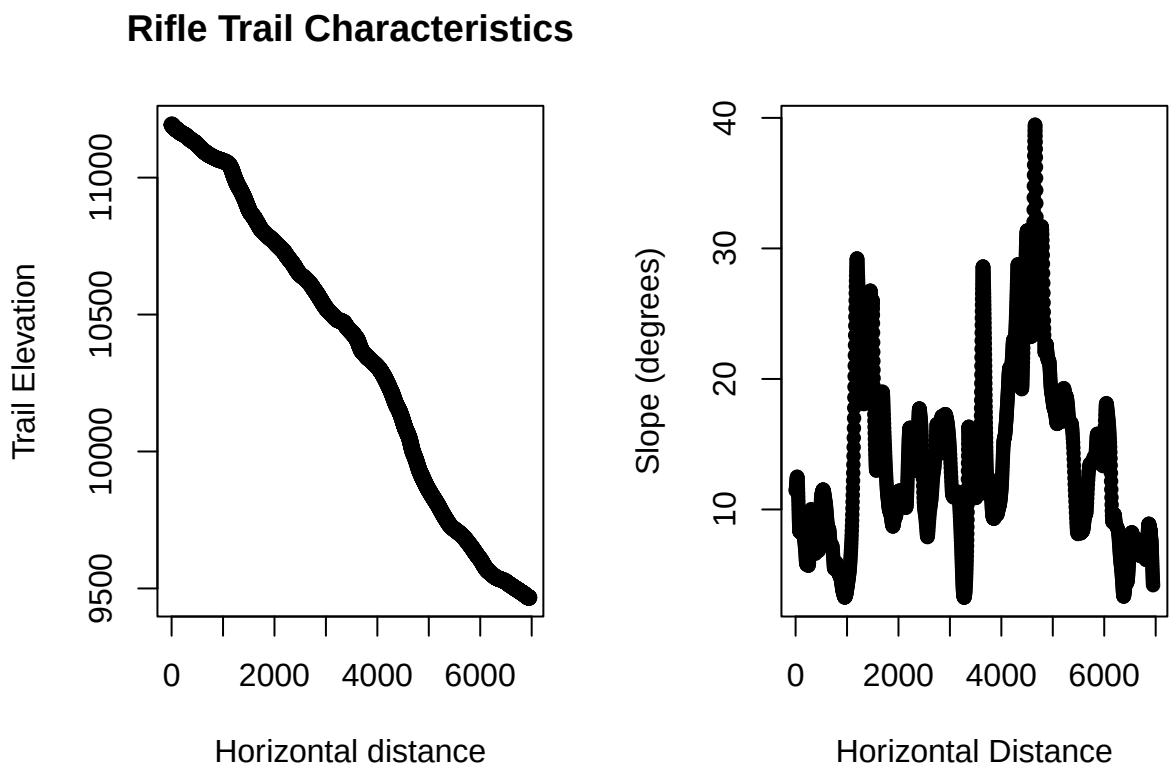


We now evaluate our fit from fastTps at the locations of the ski trail using the fit object and the predict function to find the elevations at the ski trail locations:

We will plot both elevation over distance in the x direction, and the steepness of the trailer over the x direction. Note from Dr. Nychka: "BTW: For recreational skiing, slopes over 20 degrees are considered steep and those over 30 start to get scary. For a slope of 40 or more degrees you will always get to the bottom – either skiing or sliding down if you fall!"

```
DX<- diff( rifle.trail[,1])
DY<- diff( rifle.trail[,2])
deltaDist<- sqrt( DX^2 +DY^2)
# add a zero so is the same length as locations
dist.trail<- cumsum( c(0, sqrt( DX^2 +DY^2) ))

fHat <- predict(obj, rifle.trail)

par(mfrow = c(1,2))

#plotting the elevation of the trail as a function of horizontal distances
plot(dist.trail[-1], fHat[-1],
     xlab = "Horizontal distance",
     ylab = "Trail Elevation",
     main = "Rifle Trail Characteristics")

#steepness
st<- diff( fHat)/deltaDist
```

```
#converting to degrees
stDeg <- -1*atan( st)*(360/(2*pi))

#slope plot
plot(dist.trail[-1], stDeg, pch = 16,
     xlab = "Horizontal Distance",
     ylab = "Slope (degrees)")
```

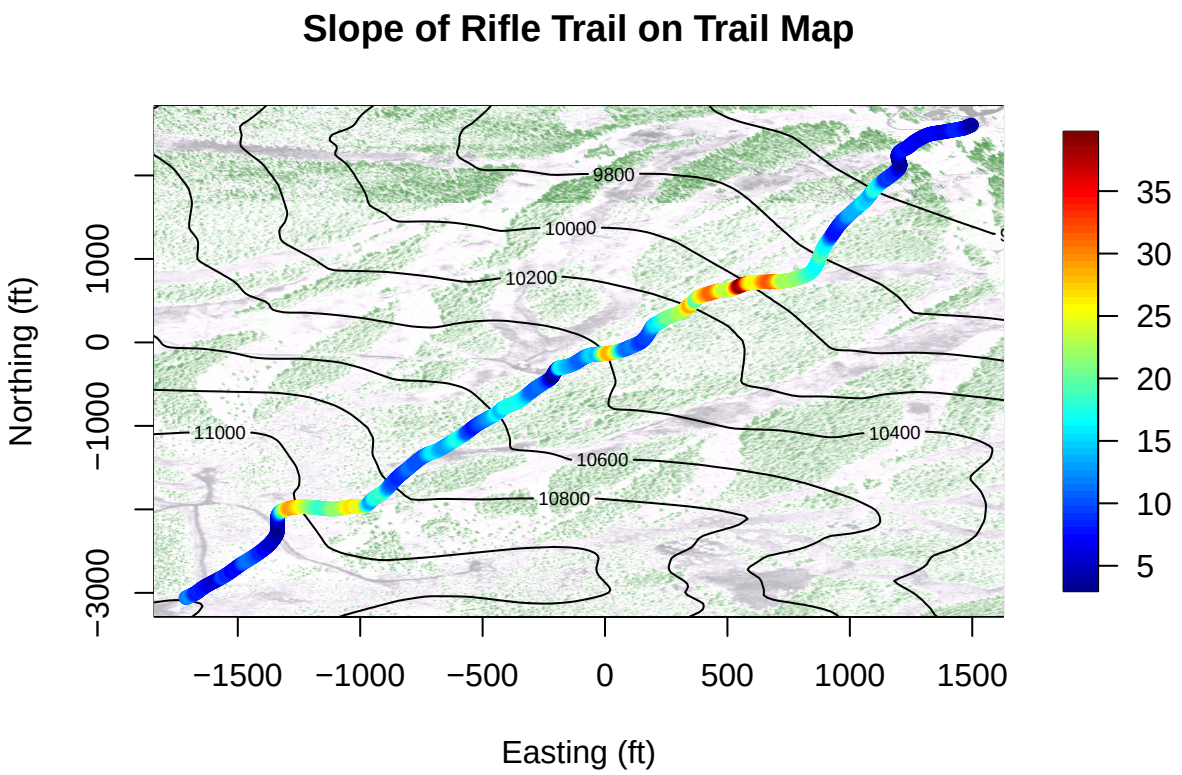



```
par(mfrow = c(1,1))
```

Now we plot the steepness of the trail on top of our image using the handy bubblePlot:

```
#putting the steepness plot on top of the map (mostly just calling plot.new)
#with the CORRECT dimensions!
bubblePlot(x=rifle.trail[-1,1],
           y=rifle.trail[-1,2],
           z=stDeg,
           col=tim.colors(),
           highlight=FALSE,
           xlab="Easting (ft)",
           ylab="Northing (ft)",
           main="Slope of Rifle Trail on Trail Map",)

#image of the map
image(rasterRifleImage,
```

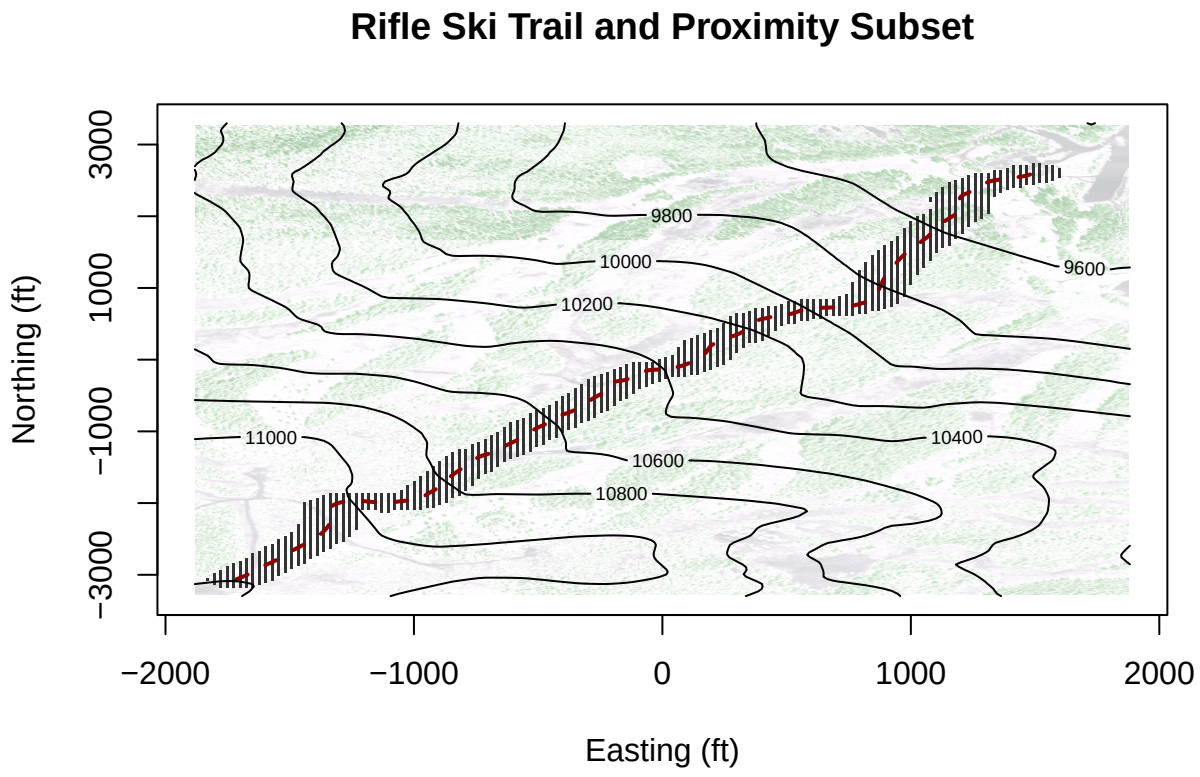
```
      col=MJ.colors(256),
      add=TRUE)

#contour on top
contour(RifleDEM, add=TRUE)

#gets overwritten so need to do it again
bubblePlot(x = rifle.trail[-1,1],
           y = rifle.trail[-1,2],
           z= stDeg, col = tim.colors(),
           highlight=FALSE, add=TRUE)
```



Let's experiment with an alternative method to avoid the Wendland type approximation. We include all grid locations within roughly 120 feet of the trail to create a subset of roughly 2000 points, and then plot them on top of the trail map:

```
n<- nrow( s)
minDist<- rep( NA,n)
for(k in 1:n){
  sTmp<- rbind(s[k,] )
  minDist[k]<- min( rdist( sTmp, rifle.trail ) )
}

ind<- which( minDist<=120)
s1<- s[ind,]
z1<- z[ind]
```

```
# image
plot( range(RifleDEM$x),range(RifleDEM$y),
      type="n",
      xlab="Easting (ft)",
      ylab="Northing (ft)",
      )
image(rasterRifleImage, col = alpha(MJ.colors(256), .5), add=TRUE)

#our subset of points
points( s1, col='grey20', pch=".", cex=.5)
#dotted line following the actual trail
lines( rifle.trail, col="red4", lty=2, lwd = 2)
#contour on top
contour(RifleDEM, add=TRUE)
title("Rifle Ski Trail and Proximity Subset")
```



First, we fit a spatial model to this subset of points with a few assumptions already being made (Matern covariance function, smoothness =1, aRange parameter = 300).

```
#really enjoying my sped-up R using MKL.
#this fit happens roughly 20 times faster with the speedup
subFit <- spatialProcess(s1, z1, aRange = 300, smoothness = 1.0)
subFit
```

```
## CALL:
```

```
## spatialProcess(x = s1, y = z1, aRange = 300, smoothness = 1)
##
##  SUMMARY OF MODEL FIT:
##
##  Number of Observations:                    1949
##  Degree of polynomial in fixed part:        1
##  Total number of parameters in fixed part:  3
##  sigma Process stan. dev:                   8.702
##  tau  Nugget stan. dev:                     4.128e-06
##  lambda   tau^2/sigma^2:                    2.251e-13
##  aRange parameter (in units of distance):   300
##  Approx.  degrees of freedom for curve      1939
##     Standard Error of df estimate:          15.32
##  log Likelihood:                            -2364.73359034774
##  log Likelihood REML:                       -2381.79402688138
##
##  ESTIMATED COEFFICIENTS FOR FIXED PART:
##
##      estimate       SE pValue
## d1 10290.0000 2.833000      0
## d2    -0.3048 0.006269      0
## d3    -0.1463 0.003775      0
##
##  COVARIANCE MODEL: stationary.cov
##   Covariance function:  Matern
##    Non-default covariance arguments and their values
## Covariance :
## [1] "Matern"
## aRange :
## [1] 300
## smoothness :
## [1] 1
## onlyUpper :
## [1] FALSE
## distMat :
## [1] NA
## Nonzero entries in covariance matrix  3798601
##
## SUMMARY FROM Max. Likelihood ESTIMATION:
## Parameters found from optim:
##       lambda
## 2.250891e-13
## Approx. confidence intervals for MLE(s)
##        lower95% upper95%
## lambda        0      Inf
##
##  Note: MLEs for  tau and sigma found analytically from lambda
##
## Summary from estimation:
## lnProfileLike.FULL lnProfileREML.FULL        lnLike.FULL        lnREML.FULL
##      -2.364734e+03      -2.381794e+03                 NA                 NA
##             lambda                tau             sigma2             aRange
##       2.250891e-13       4.128326e-06       7.571703e+01       3.000000e+02
##             eff.df                GCV
```

```
##       1.939100e+03       6.567637e-18
```

```
#gathering the measurement error/nugget
subFit_tau <- subFit$summary["tau"]
subFit_tau
```

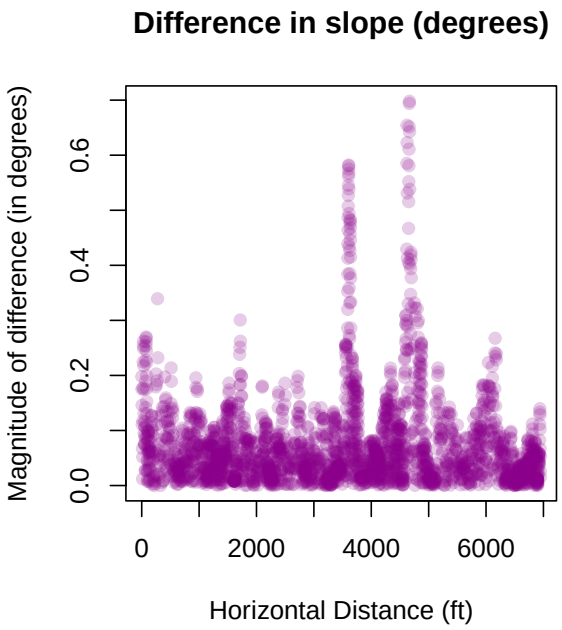
```
##          tau
## 4.128326e-06
```

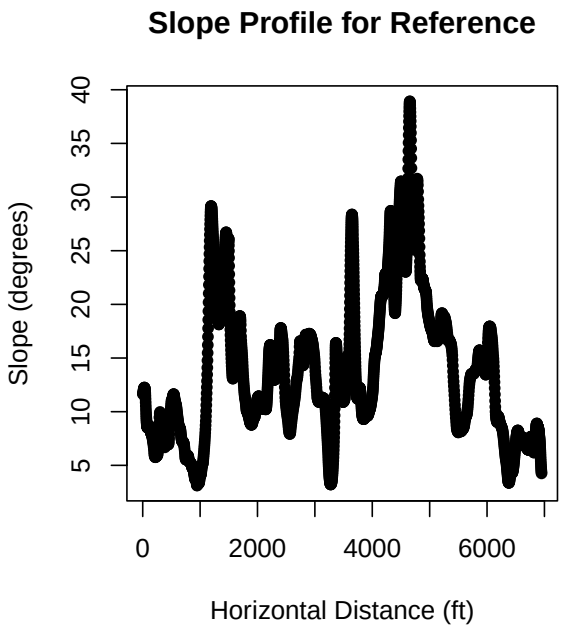
Conspiracy:

The measurement error for this spatial fit is almost zero, which lines up with our early visual estimate of a very low nugget variance (meas error) when observing the variograms. This also points towards a bit of a conspiracy: there is a chance that the original data was not taken at a very high resolution, and then filled in with some spatial process. This is consistent with the fact that the variogram has extremely nice exponential or quadratic tendencies (changes based on the radius of distance that we provide).

Now we plot the difference between the estimated slope from this model and the slope from the fit with the Wendland type approximation (see the earlier bubble plot):

```
#predcting again
fHat1 <- predict(subFit, rifle.trail)

#getting slope and turning it into degrees
st1 <- diff( fHat1)/deltaDist
stDeg1 <- (-1*atan( st1)*(360/(2*pi)))

#adding a zero for the lengths to match up
slopeDiff <- abs(stDeg - stDeg1)

par(mfrow = c(1,2))

#plotting the differences
plot(dist.trail[-1], slopeDiff, pch = 19,
     xlab = "Horizontal Distance (ft)",
     ylab = "Magnitude of difference (in degrees)",
     main = "Difference in slope (degrees)", col = alpha("darkmagenta", 0.2))

#plotting the steepness for comparison (using our new fit)
plot(dist.trail[-1], stDeg1, pch = 16,
     xlab = "Horizontal Distance (ft)",
     ylab = "Slope (degrees)", main = "Slope Profile for Reference")
```






```
par(mfrow = c(1,1))
mean(slopeDiff)
```

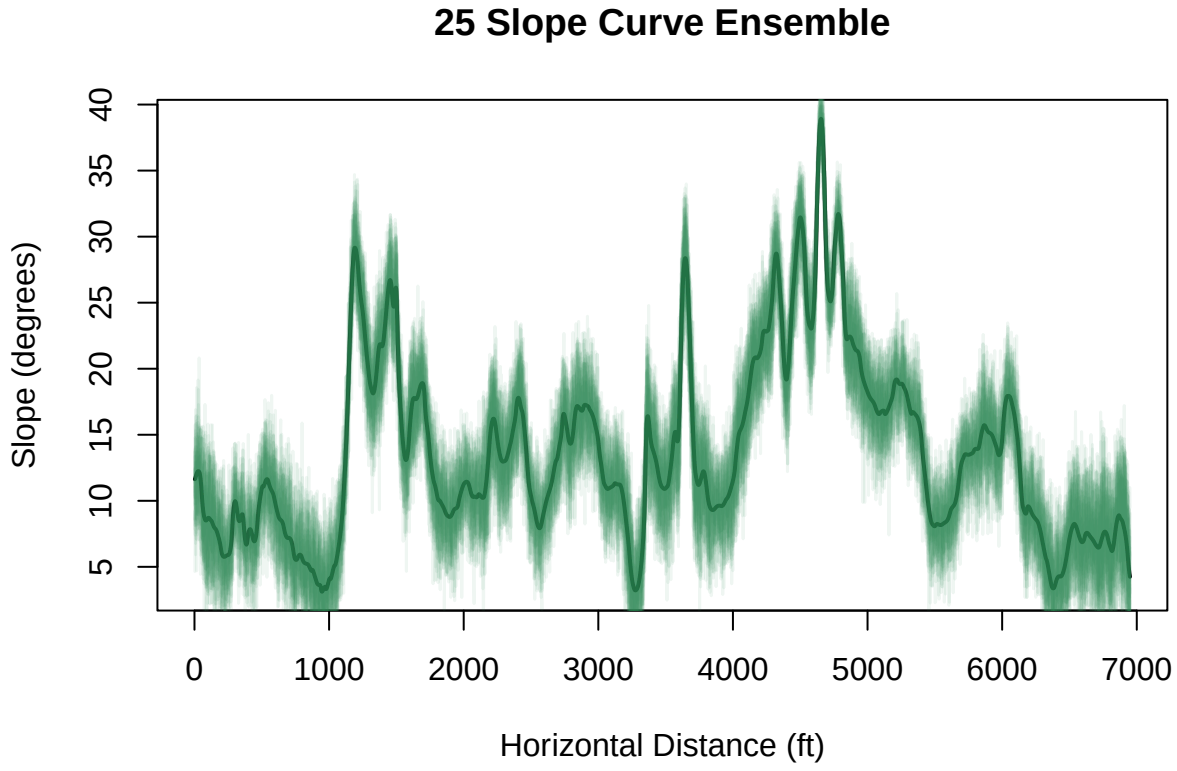
```
## [1] 0.07450908
```

The greatest differences happen when there is an extremely steep slope along the trail, although overall the difference in the two fits is minimal. At spots on the trail with a scarily steep slope, the difference between the two fits can jump up to 0.6 degrees or so, while the average discrepancy appears to be 0.075 degrees.

Now we use conditional simulation (drawing from the multivariate normal) to create an ensemble of 25 slope curves as a function of the ski trail distance and plot them along with the estimated curve.

```
#using sim spatial process for 25 draws
set.seed(777)
mnDraws <- 25
condsim <- sim.spatialProcess(subFit,
                              xp = rifle.trail,
                              M = mnDraws)

#quick function for getting slope (should've written this way earlier)
slopeFunc <- function(predictions){

  stTEMP <- diff(predictions)/deltaDist
  stDegTEMP <- -1*atan(stTEMP)*(360/(2*pi))
}

#our ensemble of slopes
ensemble <- slopeFunc(condsim)
```

```r
#plotting
plot(dist.trail[-1],
     stDeg1,
     col="black",
     type="l",
     lwd=2,
     xlab="Horizontal Distance (ft)",
     ylab="Slope (degrees)",
     main= "25 Slope Curve Ensemble")

matlines(x=dist.trail[-1],
         y=ensemble,
         type="l",
         lty=1,
         lwd = 1.5,
         col=alpha("seagreen", 0.08))
```



Above is a pretty good estimate of a range of slope profiles for the Rifle trail!