Geostatistics in East Coast Sediment Data

Larny Lopez

Packages

```
library(gstat)
library(geoR)
library(maps)
library(sp)
library(car)
library(dplyr)
```

Reading Data

```
esc = read.csv('EastCoastSediment.csv')
```

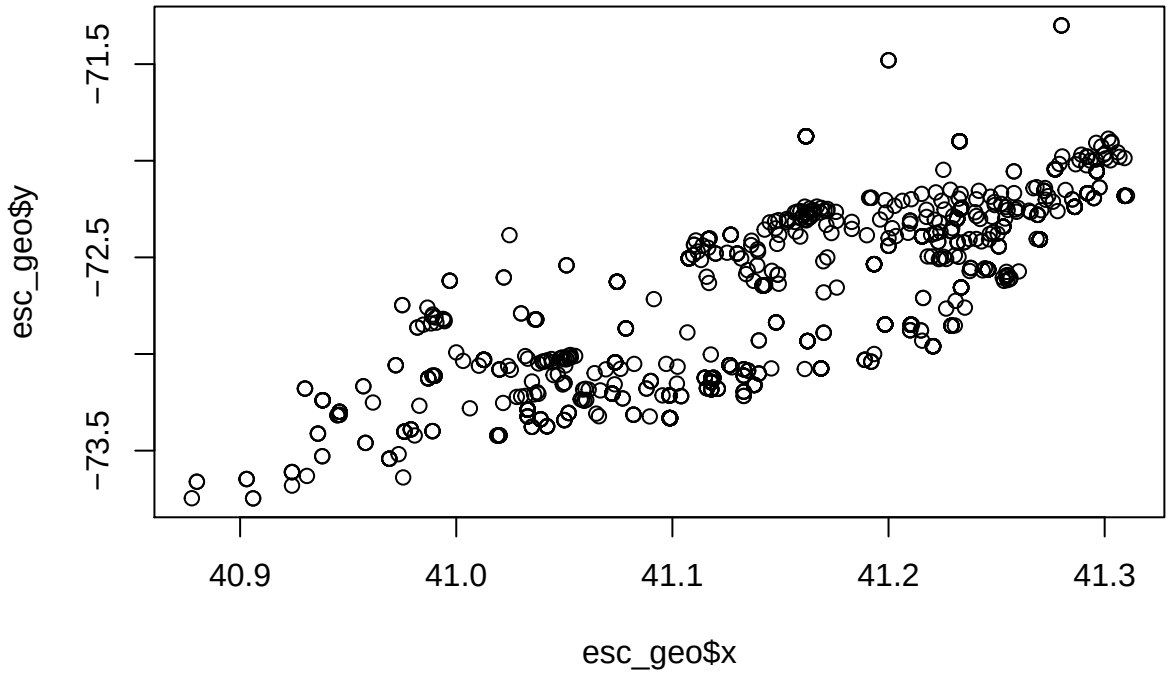
Choosing Subset and Data Cleaning

```
# Show top 25 highest frequency of areas
head(sort(table(esc$AREA), decreasing = TRUE), n = 25)
```

```
##
##                  MASSACHUSETTS                  GEORGES BANK
##                           1653                          1648
##                   LAKE MICHIGAN              LONG ISLAND SOUND
##                           1240                          1219
##                Stellwagen Bank                  GULF OF MEXICO
##                           1138                          1133
##                STELLWAGEN BANK                       MASS BAY
##                           1079                           950
##                      LOUISIANA              Long Island Sound
##                            712                           698
##                 NORTH CAROLINA                    PUERTO RICO
##                            694                           508
##            HUDSON SHELF VALLEY                  LYDONIA CANYON
##                            500                           482
##                        MASSBAY LONG ISLAND SOUND (Connecticut)
##                            470                           446
##                     NEW JERSEY                       LAKE MEAD
##                            429                           334
##                     LAKE BAIKAL          COASTAL MASSACHUSETTS
##                            320                           319
##    BUZZARDS BAY/VINEYARD SOUND                  SOUTH CAROLINA
##                            302                           286
##              BLOCK ISLAND SOUND                  NEW YORK BIGHT
##                            231                           230
```

```
##                          Maine
##                            227
```

```
# Subset Long Island Sound area and the coordinates and levels that will be used
esc_geo = esc %>% filter(AREA == "LONG ISLAND SOUND") %>%
  select("LATITUDE","LONGITUDE","SAND_PCT", "GRAVEL_PCT", "SILT_PCT", "CLAY_PCT")

# Rename data frame
names(esc_geo) = c("x", "y", "sand", "gravel", "silt","clay")

# Plot coordinates
plot(esc_geo$x, esc_geo$y)
```




```
summary(esc_geo)
```

```
##        x               y               sand              gravel
##  Min.   :40.88   Min.   :-73.75   Min.   :-9999.00   Min.   :-9999.00
##  1st Qu.:41.07   1st Qu.:-73.10   1st Qu.:    9.83   1st Qu.:    0.00
##  Median :41.16   Median :-72.65   Median :   54.10   Median :    0.20
##  Mean   :41.15   Mean   :-72.71   Mean   : -721.19   Mean   : -765.67
##  3rd Qu.:41.23   3rd Qu.:-72.33   3rd Qu.:   84.70   3rd Qu.:    3.05
##  Max.   :41.31   Max.   :-71.30   Max.   :  100.00   Max.   :   97.86
##       silt               clay
##  Min.   :-9999.00   Min.   :-9999.000
##  1st Qu.:    2.18   1st Qu.:    0.675
##  Median :   16.39   Median :    5.370
##  Mean   : -745.17   Mean   : -759.947
##  3rd Qu.:   50.87   3rd Qu.:   19.525
##  Max.   :   89.29   Max.   :   71.890
```

```
dim(esc_geo)
```

```
## [1] 1219    6
```

```r
# remove any missing values and negative/zero values
# coordinates are fine to have negative and zero
esc_complete = esc_geo %>% filter( !(sand <= 0 | gravel <= 0 | clay <= 0) )

summary(esc_complete)
```

```
##        x               y               sand           gravel
##  Min.   :40.88   Min.   :-73.75   Min.   : 0.45   Min.   : 0.010
##  1st Qu.:41.07   1st Qu.:-73.16   1st Qu.:24.54   1st Qu.: 0.230
##  Median :41.15   Median :-72.85   Median :62.52   Median : 1.490
##  Mean   :41.14   Mean   :-72.76   Mean   :55.44   Mean   : 8.275
##  3rd Qu.:41.23   3rd Qu.:-72.38   3rd Qu.:85.26   3rd Qu.: 8.098
##  Max.   :41.31   Max.   :-71.30   Max.   :99.61   Max.   :97.860
##       silt             clay
##  Min.   : 0.02   Min.   : 0.01
##  1st Qu.: 3.75   1st Qu.: 1.19
##  Median :16.59   Median : 5.57
##  Mean   :25.36   Mean   :10.92
##  3rd Qu.:49.19   3rd Qu.:18.83
##  Max.   :87.53   Max.   :52.24
```

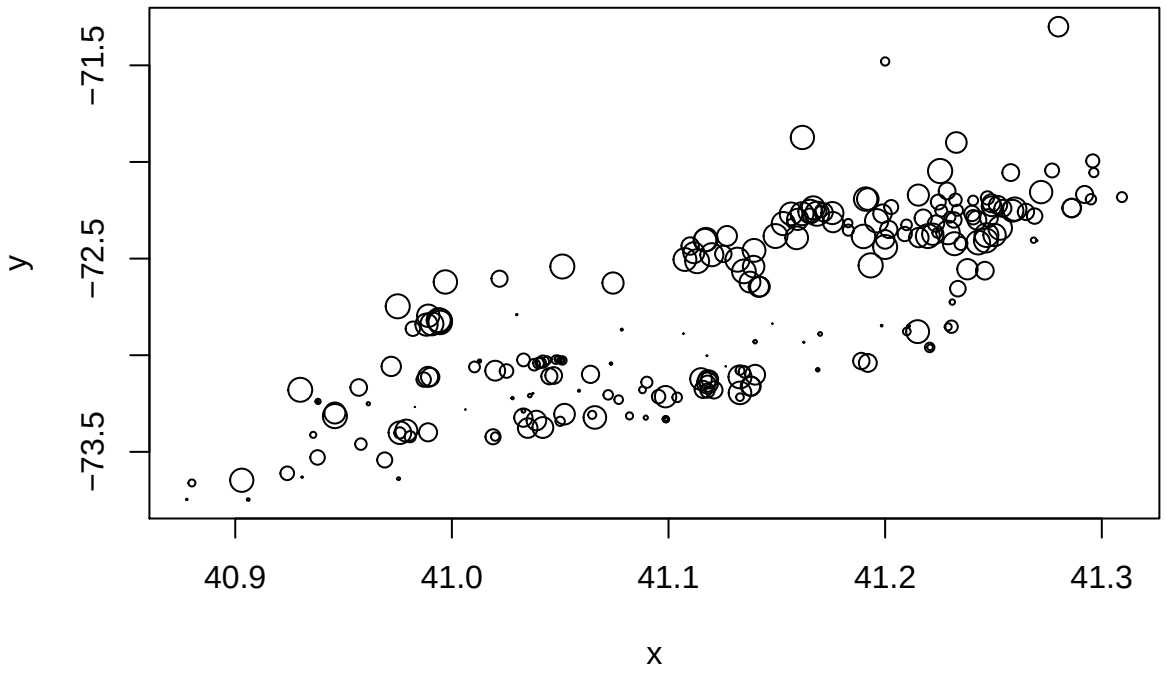
```r
dim(esc_complete)
```

```
## [1] 790   6
```

```r
# convert x and y to coordinates to be able to remove duplicates
coordinates(esc_complete) = ~x+y
esc_rmdup = remove.duplicates(esc_complete)

dim(esc_rmdup)
```

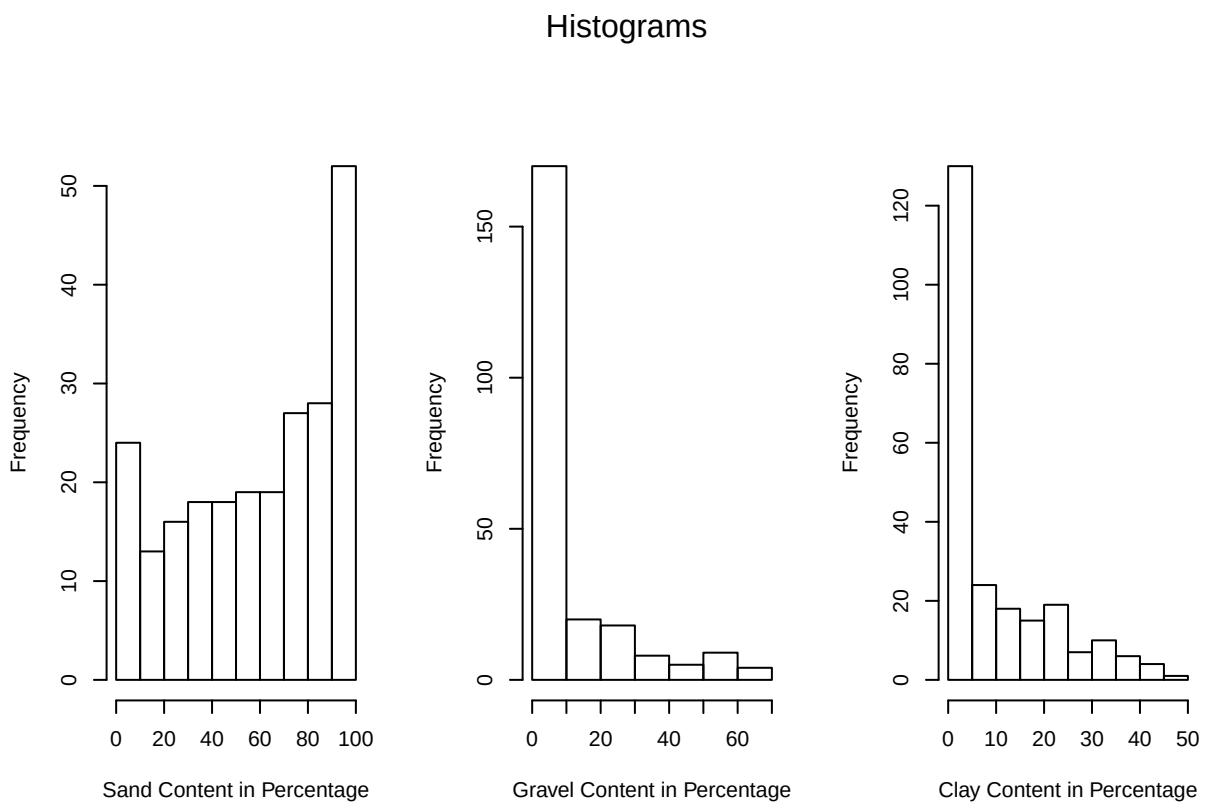
```
## [1] 234   4
```

```r
# return back to df
esc_df = as.data.frame(esc_rmdup)

# plot coordinates of final complete data frame
plot(esc_df$x, esc_df$y, cex = esc_df$sand / mean(esc_df$sand), xlab = "x", ylab = "y")
```

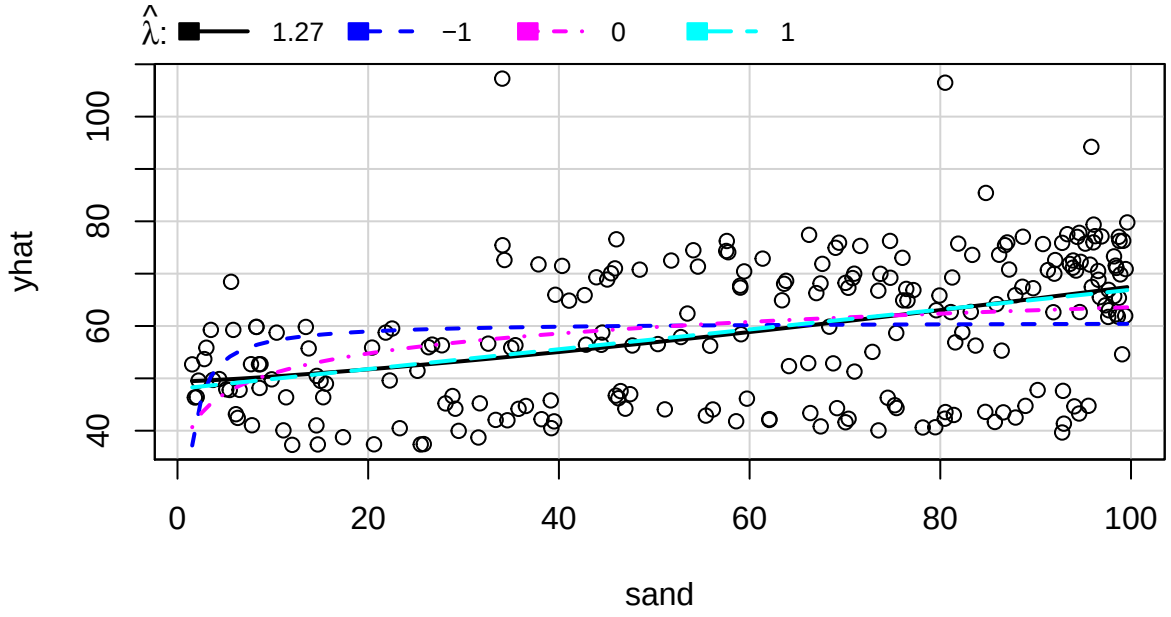


Exploratory Analysis

```
par(mfrow = c(1,3), oma = c(0, 0, 3, 0))
hist(esc_df$sand, main = NA, xlab = "Sand Content in Percentage")
hist(esc_df$gravel, main = NA, xlab = "Gravel Content in Percentage")
hist(esc_df$clay, main = NA, xlab = "Clay Content in Percentage")
mtext("Histograms", outer = TRUE)
```



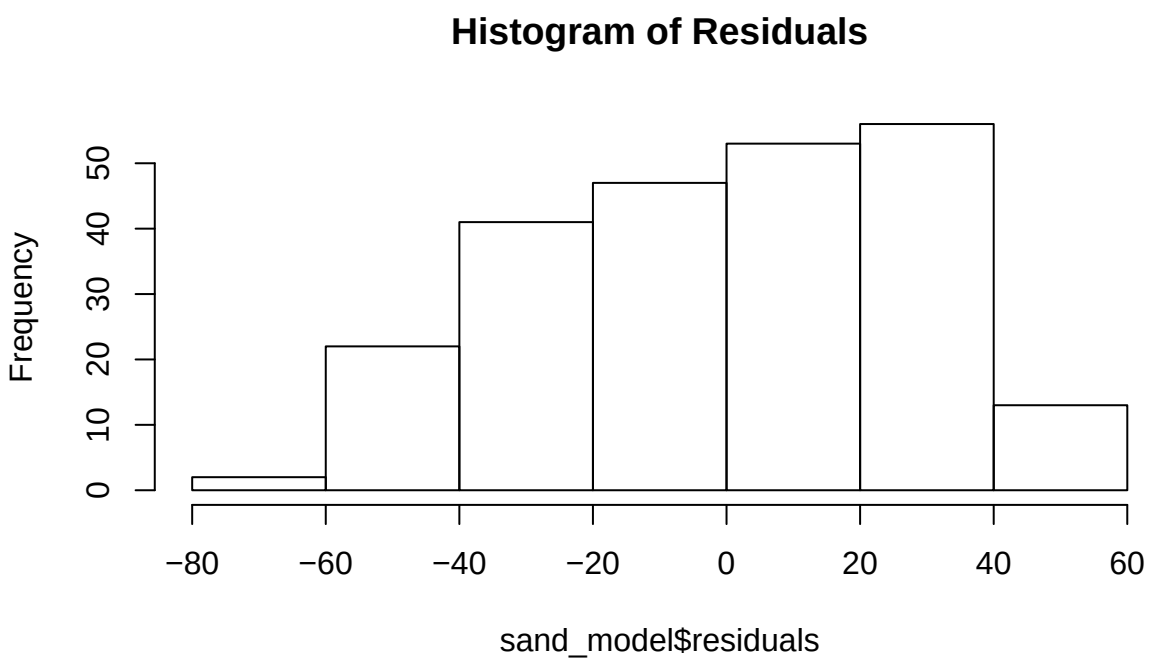

```
# check residual histogram after transformation (for universal kriging)
par(mfrow = c(1,1))
sand_model = lm(sand ~ x + y, data = esc_df)
inverseResponsePlot(sand_model)
```




```
##      lambda      RSS
## 1  1.272684 34438.83
## 2 -1.000000 40379.56
## 3  0.000000 36447.72
```

```
## 4  1.000000 34493.30
```

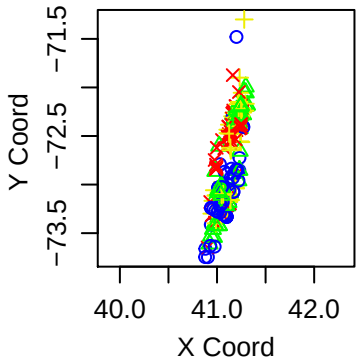
```
hist(sand_model$residuals, main = "Histogram of Residuals")
```

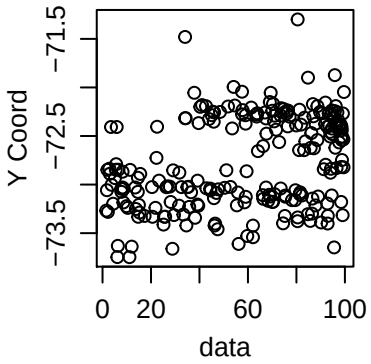


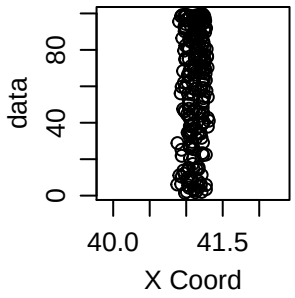

```
####### lambda = 1, no transformation #########

# convert to geodata to see summary, plot, and directional variogram
esc_geodata = as.geodata(esc_df)
summary(esc_geodata)
```

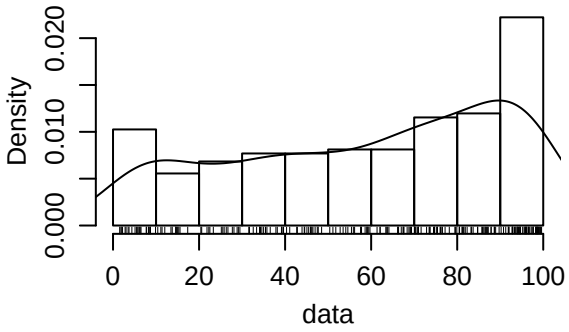
```
## Number of data points: 234
##
## Coordinates summary
##            x       y
## min 40.87766 -73.747
## max 41.30933 -71.300
##
## Distance summary
##       min       max
## 0.0000017 2.4791657
##
## Data summary
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##  1.54000 34.14000 66.30500 59.19632 87.17000 99.61000
```

```
plot(esc_geodata)
```



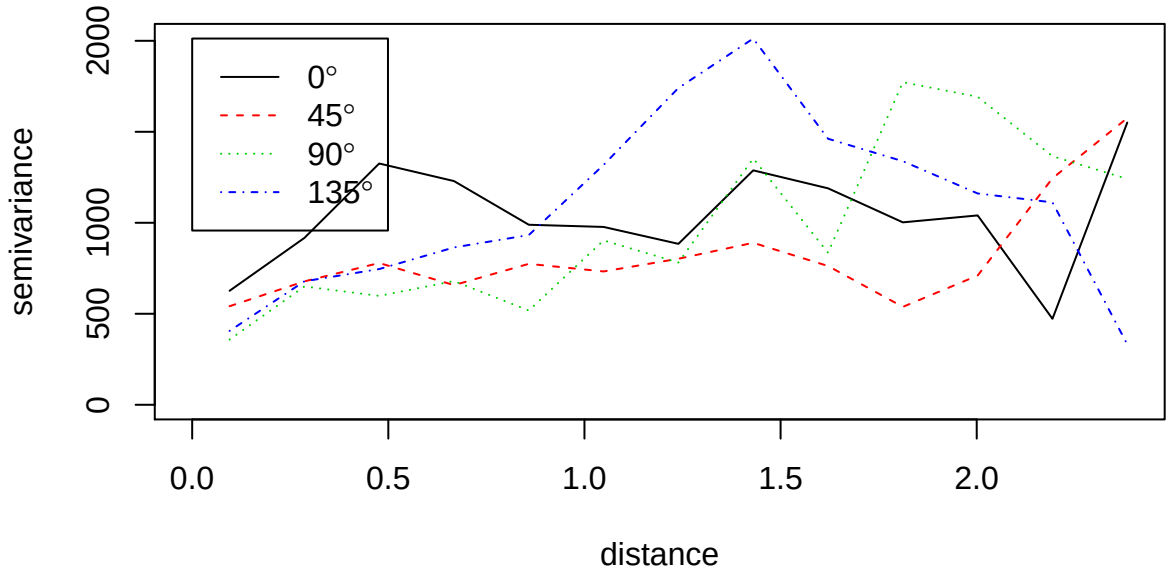






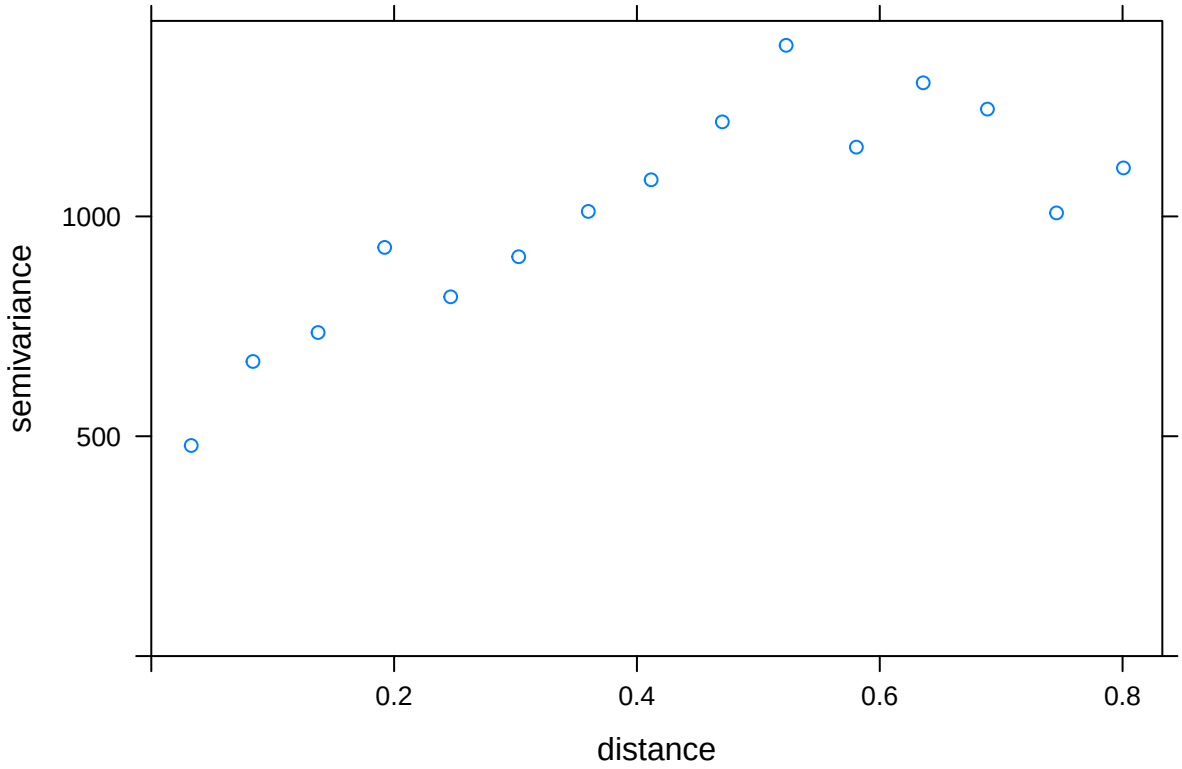

```
plot(variog4(esc_geodata))
```

```
## variog: computing variogram for direction = 0 degrees (0 radians)
##         tolerance angle = 22.5 degrees (0.393 radians)
## variog: computing variogram for direction = 45 degrees (0.785 radians)
##         tolerance angle = 22.5 degrees (0.393 radians)
## variog: computing variogram for direction = 90 degrees (1.571 radians)
##         tolerance angle = 22.5 degrees (0.393 radians)
## variog: computing variogram for direction = 135 degrees (2.356 radians)
##         tolerance angle = 22.5 degrees (0.393 radians)
## variog: computing omnidirectional variogram
```

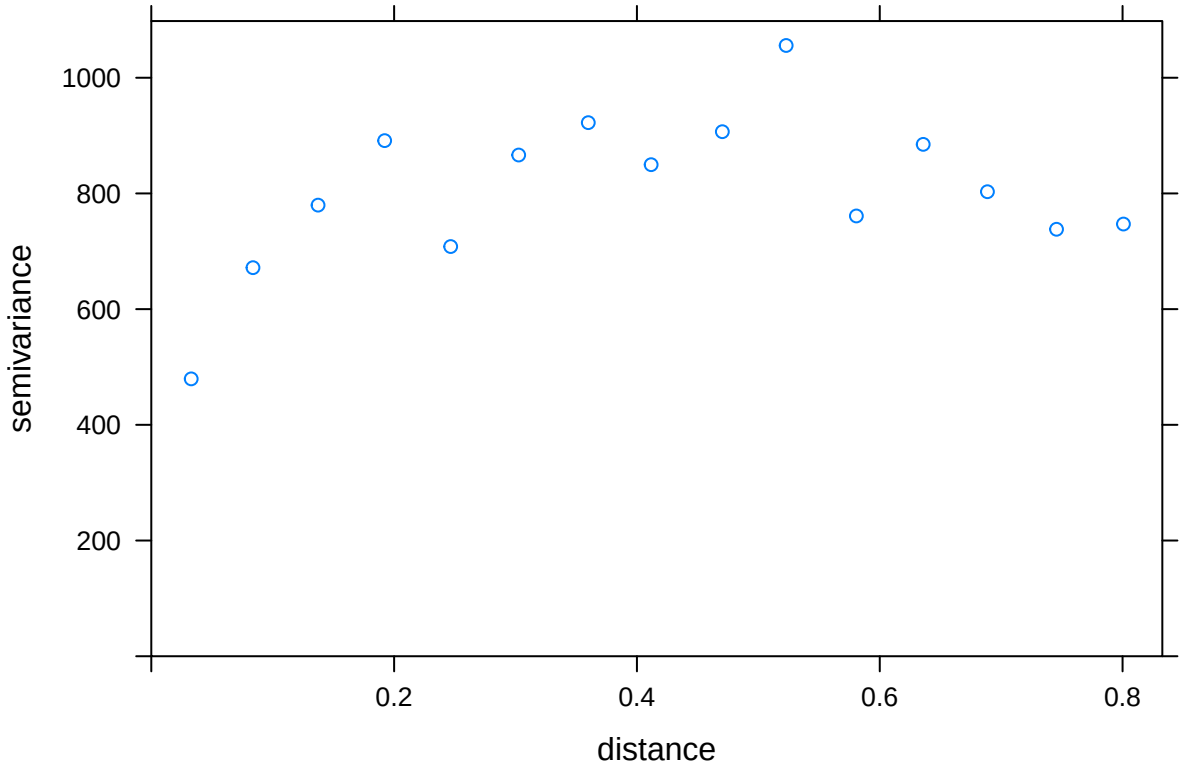


Choosing Variogram Model

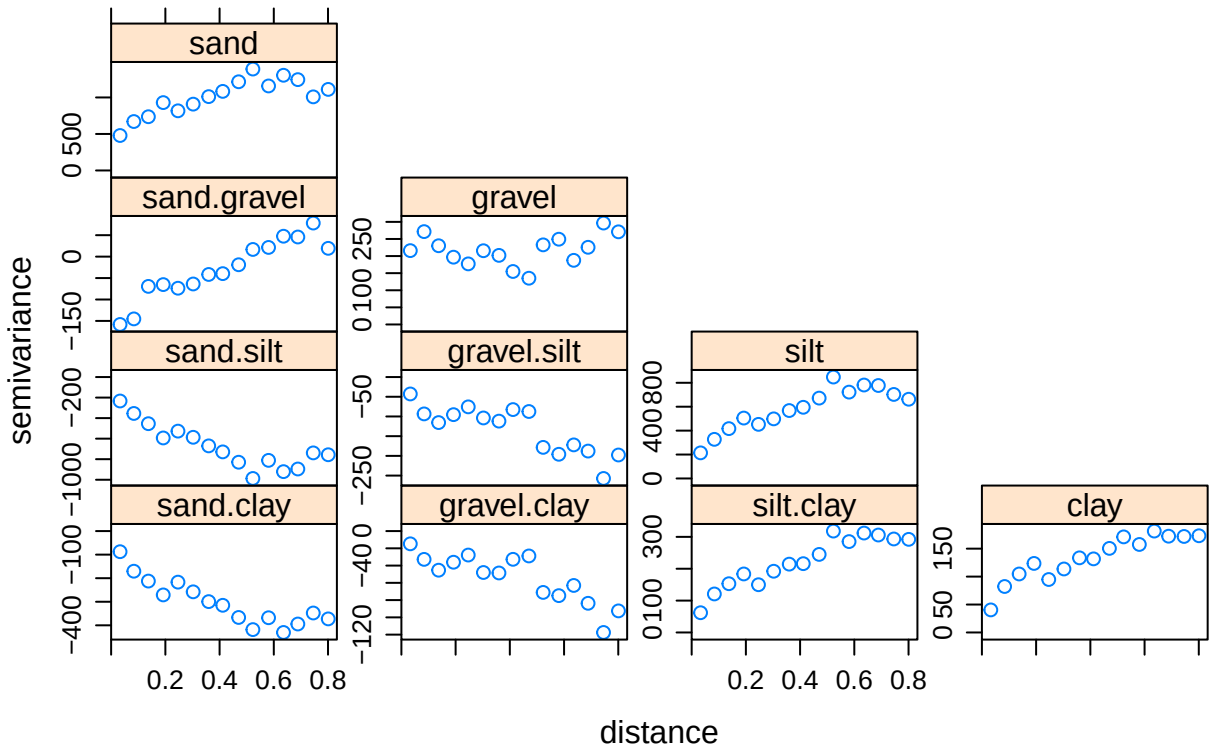
```
# No Trend
esc_gstat_ok = gstat(id = "sand", formula = sand ~ 1,
                     locations = ~ x + y, data = esc_df)

# With 1st order trend
esc_gstat_uk = gstat(id = "sand", formula = sand ~ x + y,
                     locations = ~ x + y, data = esc_df)

# For co kriging
esc_gstat_ck = gstat(id = "sand", formula = sand ~ 1,
                     locations = ~ x + y, data = esc_df)
esc_gstat_ck = gstat(esc_gstat_ck, id = "gravel", formula = gravel ~ 1,
                     locations = ~ x + y, data = esc_df)
esc_gstat_ck = gstat(esc_gstat_ck, id = "silt", formula = silt ~ 1,
                     locations = ~ x + y, data = esc_df)
esc_gstat_ck = gstat(esc_gstat_ck, id = "clay", formula = clay ~ 1,
                     locations = ~ x + y, data = esc_df)

# variograms
esc_vario_ok = variogram(esc_gstat_ok)
esc_vario_uk = variogram(esc_gstat_uk)
esc_vario_ck = variogram(esc_gstat_ck)

plot(esc_vario_ok)
```



```
plot(esc_vario_uk)
```

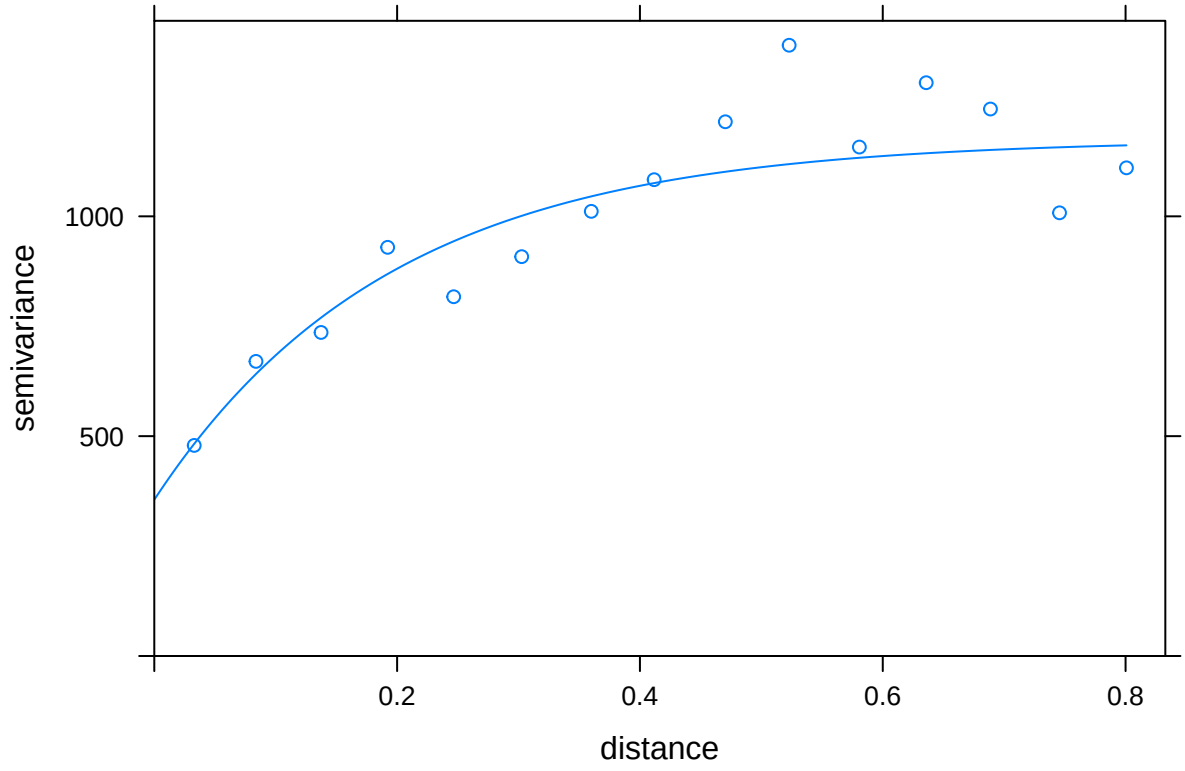



```
plot(esc_vario_ck)
```

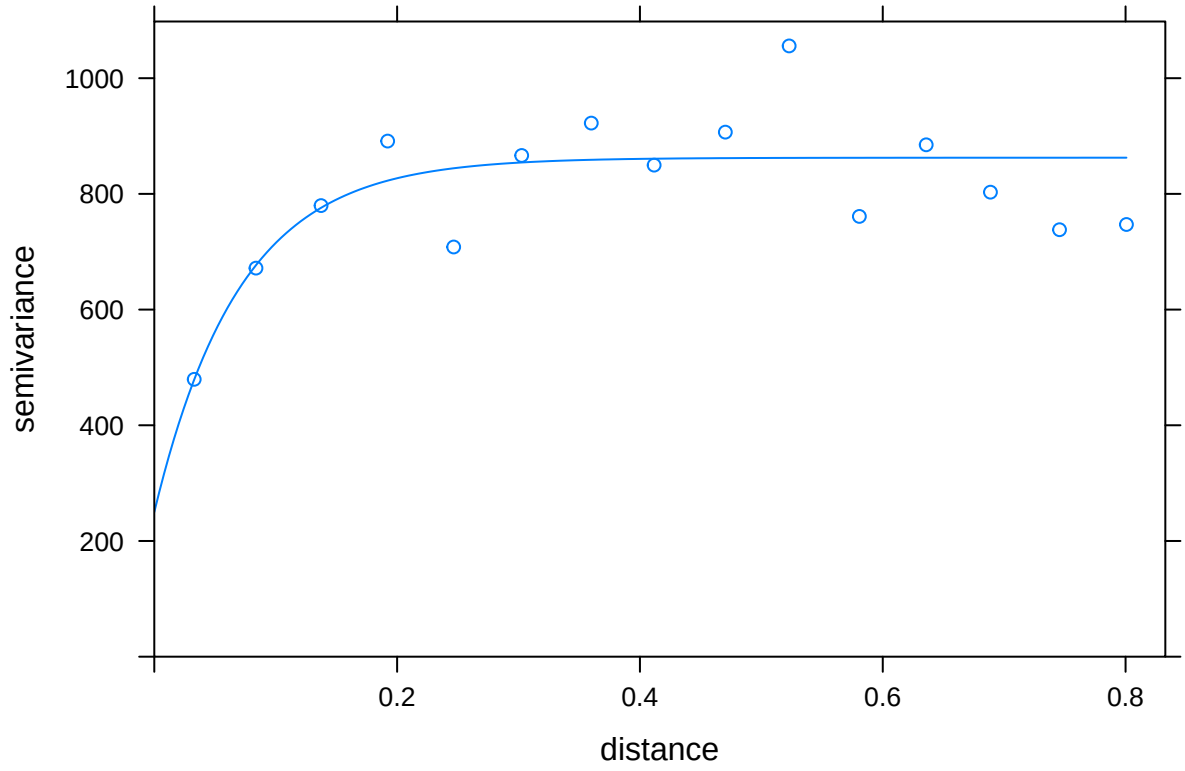



```
fit_ok = fit.variogram(esc_vario_ok, vgm(900, "Exp", 0.6, 500))
fit_uk = fit.variogram(esc_vario_uk, vgm(400, "Exp", 0.4, 500))
```

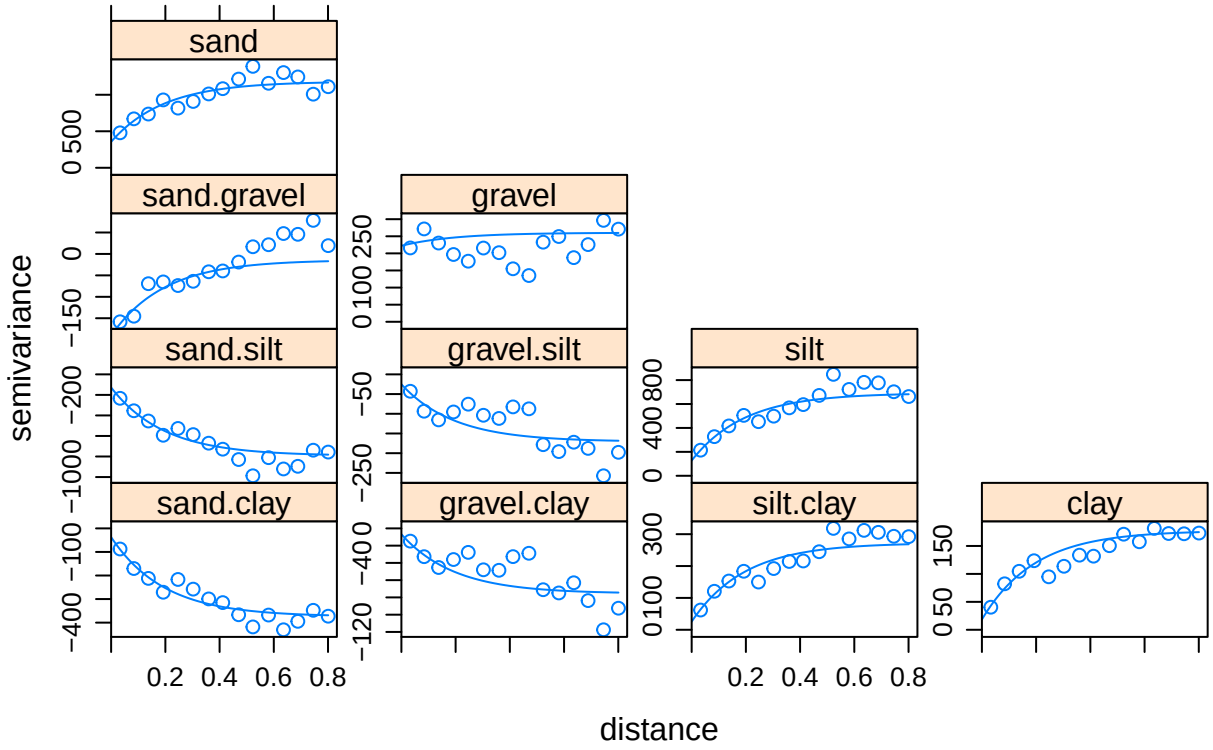
```
fit_ck = fit.lmc(esc_vario_ck, esc_gstat_ck, model = fit_ok)

plot(esc_vario_ok, fit_ok)
```




```
plot(esc_vario_uk, fit_uk)
```



```
plot(esc_vario_ck, fit_ck)
```




```
cv_ok = krige.cv(sand ~ 1, data = esc_df,
                 locations = ~ x + y, model = fit_ok, nfold = nrow(esc_df))
sum(cv_ok$residual^2)
```

```
## [1] 106437.3
```

```
cv_uk = krige.cv(sand ~ x + y, data = esc_df,
                 locations = ~ x + y, model = fit_uk, nfold = nrow(esc_df))
sum(cv_uk$residual^2)
```

```
## [1] 99945.52
```

```
##### Universal has lower PRESS than Ordinary Kriging#################


# cv_ck = gstat.cv(fit_ck)

# ERROR from running gstat.cv#
# 0%
#non-positive definite coefficient matrix in structure 2
#No Intrinsic Correlation or Linear Model of Coregionalization found
# Reason: coefficient matrix not positive definite[add `set = list(nocheck = 1)'
# to the gstat() or krige() to ignore the following error]
# Error in predict.gstat(object, newdata = data[sel, ], ...) :
#   value not allowed for: variograms do not satisfy a legal model
```

Raster Map for Predicted Values and Variance

```
x_range = range(esc_df$x)
y_range = range(esc_df$y)
x_range; y_range
```

```
## [1] 40.87766 41.30933
```

```
## [1] -73.747 -71.300
```

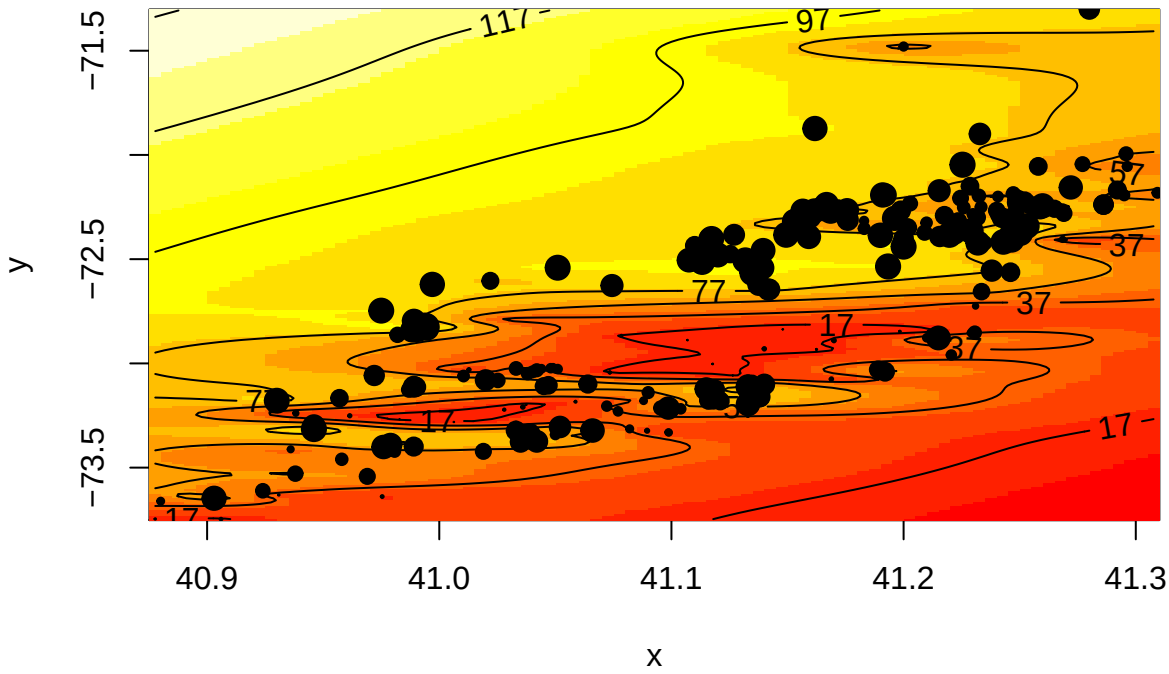
```
x_seq = seq(x_range[1], x_range[2], by = 0.005)
y_seq = seq(y_range[1], y_range[2], by = 0.01)

grd = expand.grid(x = x_seq, y = y_seq)

# plot(grd)

krige_uk = krige(id = "sand", sand ~ x + y,
                 locations = ~ x + y, model = fit_uk, data = esc_df, newdata =grd)
```

```
## [using universal kriging]
```

```
mat_pred = matrix(krige_uk$sand.pred, length(x_seq), length(y_seq))

range(krige_uk$sand.pred)
```

```
## [1]  -2.753116 138.191717
```

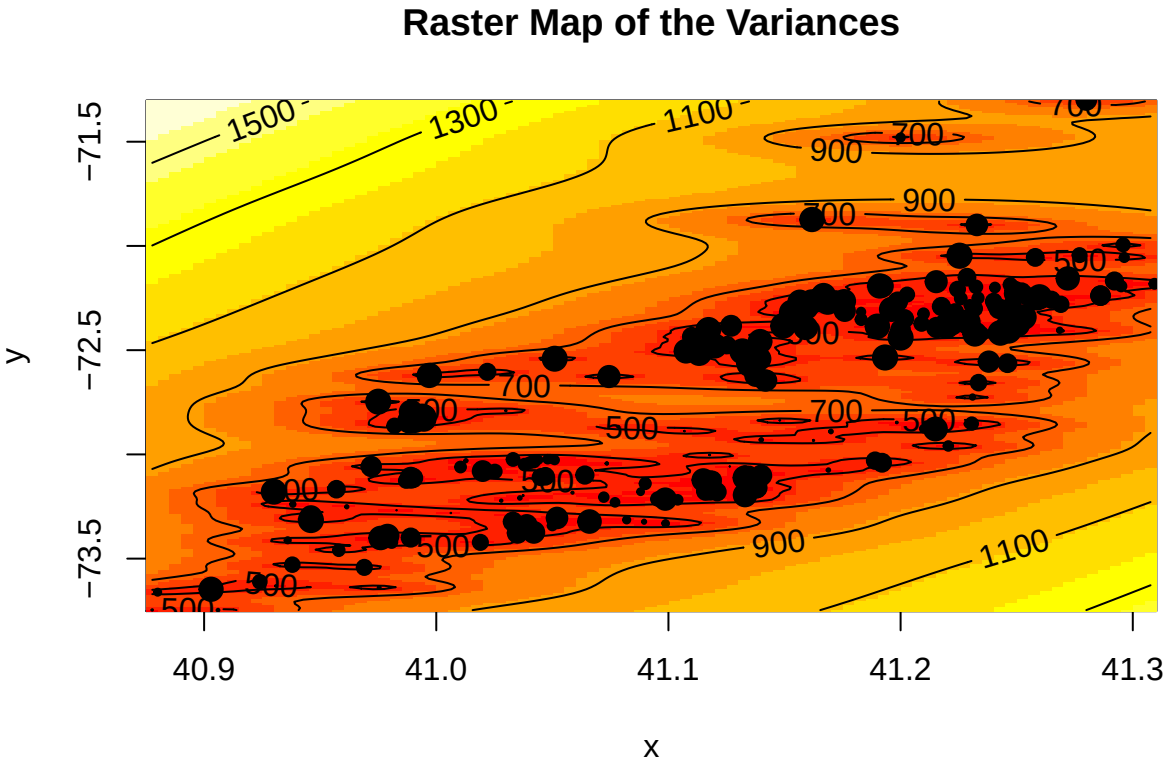
```
image(x_seq, y_seq, mat_pred, xlab = "x", ylab = "y", main = "Raster Map of the Predicted Values")
contour(x_seq, y_seq, mat_pred, add = TRUE,
        col = "black", labcex = 1, levels = seq(-3,140, by = 20))
points(esc_df$x, esc_df$y, cex = esc_df$sand / mean(esc_df$sand), pch = 19)
```




```
mat_var = matrix(krige_uk$sand.var, length(x_seq), length(y_seq))

range(krige_uk$sand.var)
```

```
## [1]  308.1993 1679.3373
```

```
image(x_seq, y_seq, mat_var, xlab = "x", ylab = "y", main = "Raster Map of the Variances")
contour(x_seq, y_seq, mat_var, add = TRUE,
        col = "black", labcex = 1, levels = seq(300, 1680, by = 200))
points(esc_df$x, esc_df$y, cex = esc_df$sand / mean(esc_df$sand), pch = 19)
```



Checking for Anisotropy

```
# this was done before kriging
# but specifying direction was not necessary
maxd  = 0.4

var1 = variog(esc_geodata, dir=pi/2, tol=pi/4, max.dist=maxd)
```

```
## variog: computing variogram for direction = 90 degrees (1.571 radians)
##         tolerance angle = 45 degrees (0.785 radians)
```

```
var2 = variog(esc_geodata, dir=pi/2.57, tol=pi/4, max.dist=maxd)
```

```
## variog: computing variogram for direction = 70.039 degrees (1.222 radians)
##         tolerance angle = 45 degrees (0.785 radians)
```

```
var3 = variog(esc_geodata, dir=pi/3.6, tol=pi/4, max.dist=maxd)
```

```
## variog: computing variogram for direction = 50 degrees (0.873 radians)
##         tolerance angle = 45 degrees (0.785 radians)
```

```
var4 = variog(esc_geodata, dir=pi/6, tol=pi/4, max.dist=maxd)
```

```
## variog: computing variogram for direction = 30 degrees (0.524 radians)
##         tolerance angle = 45 degrees (0.785 radians)
```

```
var5 = variog(esc_geodata, dir=pi/18, tol=pi/4, max.dist=maxd)
```

```
## variog: computing variogram for direction = 10 degrees (0.175 radians)
##         tolerance angle = 45 degrees (0.785 radians)
```

```
var6 = variog(esc_geodata, dir=0.944*pi, tol=pi/4, max.dist=maxd)
```

```
## variog: computing variogram for direction = 169.92 degrees (2.966 radians)
##         tolerance angle = 45 degrees (0.785 radians)
```

```
var7 = variog(esc_geodata, dir=0.833*pi, tol=pi/4, max.dist=maxd)
```

```
## variog: computing variogram for direction = 149.94 degrees (2.617 radians)
##         tolerance angle = 45 degrees (0.785 radians)
```

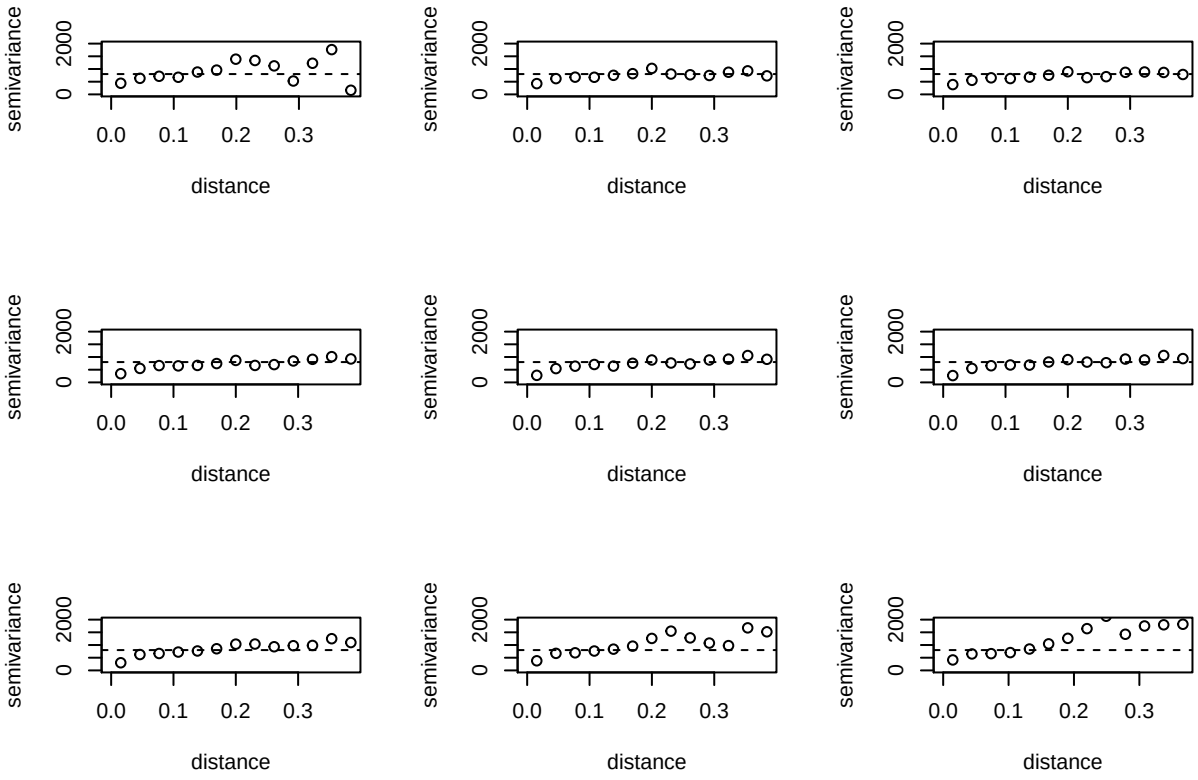
```
var8 = variog(esc_geodata, dir=0.722*pi, tol=pi/4, max.dist=maxd)
```

```
## variog: computing variogram for direction = 129.96 degrees (2.268 radians)
##         tolerance angle = 45 degrees (0.785 radians)
```

```
var9 = variog(esc_geodata, dir=0.611*pi, tol=pi/4, max.dist=maxd)
```

```
## variog: computing variogram for direction = 109.98 degrees (1.92 radians)
##         tolerance angle = 45 degrees (0.785 radians)
```

```
ylim = 2000
alpha = as.integer(800)

par(mfrow = c(3,3))
plot(var1, ylim=c(0,ylim)) ; abline(h = alpha, lty = 2)
plot(var2, ylim=c(0,ylim)) ; abline(h = alpha, lty = 2)
plot(var3, ylim=c(0,ylim)) ; abline(h = alpha, lty = 2)
plot(var4, ylim=c(0,ylim)) ; abline(h = alpha, lty = 2)
plot(var5, ylim=c(0,ylim)) ; abline(h = alpha, lty = 2)
plot(var6, ylim=c(0,ylim)) ; abline(h = alpha, lty = 2)
plot(var7, ylim=c(0,ylim)) ; abline(h = alpha, lty = 2)
plot(var8, ylim=c(0,ylim)) ; abline(h = alpha, lty = 2)
plot(var9, ylim=c(0,ylim)) ; abline(h = alpha, lty = 2)
```




```
#From the plots above approximately find the distance at which the variograms reach the value of 700:


#Compute the coordinates:
theta = c(0, pi/9, pi/4.5, pi/3, pi/2.25, pi/18, pi/6, pi/3.6, pi/2.571)
range = c(1.3, 1.6, 1.7, 1.8, 1.8, 1.8, 1.8, 1.4, 1.4)

# No big difference between the distances, relatively close to each other

x1 = cos(theta[1:5])*range[1:5]
y1 = sin(theta[1:5])*range[1:5]

x2 = range[6:9]*sin(theta[6:9])
y2 = -range[6:9]*cos(theta[6:9])

x11 = -x1
y11 = -y1

x22 = -x2
y22 = -y2

par(mfrow = c(1,1))
plot(x1,y1, xlim=c(-2,2), ylim=c(-2,2), xaxt="n", yaxt="n",
     ylab="y", xlab="x")
points(x11,y11)
points(x2,y2)
points(x22,y22)

segments(x1,y1, x11, y11)
```

```
segments(x2,y2, x22, y22)

segments(0, -3, 0, 3, lty=2)
segments(-3, 0, 3, 0, lty=2)
```